CONTRIBUTED PAPER

Deep-Learning-Based 3-D Surface Reconstruction—A Survey

This survey presents a comprehensive overview of these state-of-the-art deep-learning-based approaches to 3-D surface reconstruction.

By Anis Farshian, Markus Götz, *Member IEEE*, Gabriele Cavallaro, *Senior Member IEEE*, Charlotte Debus, *Member IEEE*, Matthias Nießner, Jón Atli Benediktsson, *Fellow IEEE*, and Achim Streit

ABSTRACT | In the last decade, deep learning (DL) has significantly impacted industry and science. Initially largely motivated by computer vision tasks in 2-D imagery, the focus has shifted toward 3-D data analysis. In particular, 3-D surface reconstruction, i.e., reconstructing a 3-D shape from sparse input, is of great interest to a large variety of application fields. DL-based approaches show promising quantitative and qualitative surface reconstruction performance compared to traditional computer vision and geometric algorithms. This survey provides a comprehensive overview of these DL-based methods for 3-D surface reconstruction. To this end, we will first discuss input data modalities, such as volumetric data, point clouds, and RGB, single-view, multiview, and depth images, along with corresponding acquisition technologies and common benchmark datasets. For practical purposes, we also discuss evaluation metrics enabling us to judge the reconstructive performance of different methods. The main part of the document will introduce a methodological taxonomy ranging from point- and mesh-based techniques to volumetric and implicit neural approaches. Recent research trends, both methodological and for applications, are highlighted, pointing toward future developments.

KEYWORDS | 3-D deep learning (DL); 3-D surface reconstruction; geometric DL; geometry processing; machine learning.

I. Introduction

In the last decade, advances in artificial intelligence, in particular in deep learning (DL) [1], [2], [3], have been adopted by a multitude of fields and have, thus, led to major breakthroughs in science and industry alike. One of the major driving forces behind these developments is the field of computer vision, and its desire to "teach" machines how to recognize patterns within image and video data. Initially, a strong emphasis was placed on the interpretation of 2-D information; however, recent advances in cost-effective scanner-based data acquisition and the establishment of large-scale shape repositories have brought the analysis of 3-D data into focus. Still, complexity, variety, and irregularities in 3-D shape representations pose significant methodological challenges.

The reconstruction of 3-D surfaces of objects from different types of input data formats, such as point clouds, depth maps, single-view, or multiview images, is fundamental to a number of application fields, such as computer vision, robotics, CAD, medicine, city planning, disaster prevention, and archeology. One of the special use cases of 3-D reconstruction is human shape reconstructions and pose estimation from images or videos, which is addressed by some other works [4], [5].

Manuscript received 1 March 2022; revised 9 June 2023 and 24 September 2023; accepted 26 September 2023. Date of publication 30 October 2023; date of current version 17 November 2023. *(Corresponding author: Markus Götz.)*

Anis Farshian, **Markus Götz**, **Charlotte Debus**, and **Achim Streit** are with the Steinbuch Centre for Computing, Karlsruhe Institute of Technology, 76344 Karlsruhe, Germany (e-mail: anis.farshian@kit.edu; markus.goetz@kit.edu; charlotte.debus@kit.edu; achim.streit@kit.edu).

Gabriele Cavallaro is with the Jülich Supercomputing Centre, Forschungszentrum Jülich, 52428 Jülich, Germany, and also with the Faculty of Industrial Engineering, Mechanical Engineering and Computer Science, University of Iceland, 107 Reykjavik, Iceland (e-mail: g.cavallaro@fz-juelich.de).

Matthias Nießner is with the Visual Computing Laboratory, Department of Informatics, Technical University of Munich, 80333 Munich, Germany (e-mail: niessner@tum.de).

Jón Atli Benediktsson is with the Faculty of Electrical and Computer Engineering, University of Iceland, 102 Reykjavik, Iceland (e-mail: benedikt@hi.is).

Digital Object Identifier 10.1109/JPROC.2023.3321433

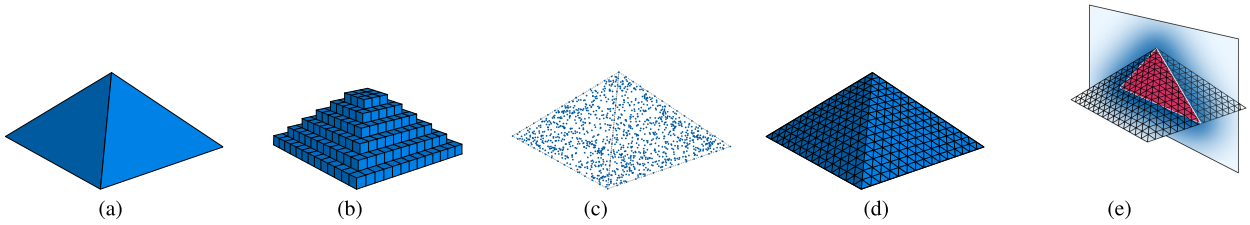


Fig. 1. *Output representations of various 3-D surface reconstruction approaches. DL-based 3-D surface reconstruction approaches can be broadly classified into four main categories according to their representation: volumetric, point cloud, mesh, and an example of implicit neural representation based on SDF. (a) Object. (b) Voxelized. (c) Point cloud. (d) Mesh. (e) Implicit.*

Despite a long research history for 3-D surface reconstruction, the precise representation of 3-D geometrical objects remains an unsolved problem, usually requiring the reconstructed 3-D surfaces to be: 1) highly resolved and smooth; 2) water-tight, i.e., "without gaps"; 3) in accordance with possible ground truth; 4) robust against noisy or incomplete input; and 5) simultaneously, densely, and compressibly represented.

Classical approaches for addressing these problems encompass geometrical or simplistic machine-learning-based algorithms [6], [7]. Most of these methods are not able to comprehensively and consistently reconstruct arbitrary detailed 3-D surfaces. Well-known techniques, such as (screened) Poisson surface reconstruction (PSR) [8], [9], the ball-pivoting algorithm (BPA) [10], and Delaunay triangulation [11], [12], still suffer from scalability issues and struggle to reconstruct fine details for large-scale data.

The recent successes of deep neural networks (DNNs) in other data-driven computational problems, such as classification [13], [14], object detection [14], [15], and segmentation [13], [14], [16], have sparked interest in utilizing DL for 3-D surface reconstruction. Partially overlapping with the latter is the task of shape completion, i.e., enhancing the input data with (partially) occluded shape information.

Several reconstruction-related surveys [17], [18] present early approaches, with [17] providing an overview of the classical and non-DL-based surface reconstruction methods from point clouds with respect to priors and [18] reviewing RGB-D scene reconstruction approaches. There is another DL-based surface reconstruction survey [19] with a focus on image-based methods. This article, however, covers broader data modalities and reviews recent trends in 3-D surface reconstruction including implicit neural representation and neural radiance fields (NeRFs) thoroughly.

Therefore, the fast-paced development of the field makes it, however, necessary to revisit up-to-date research frequently. The current landscape of DL-based 3-D reconstruction can be broadly classified into four main categories according to their representation, as depicted in Fig. 1: 1) volumetric, i.e., representing a surface with small cuboids, either a dense 3-D voxel grid [20], [21], [22], [23], [24], [25] or an octree [26], [27], [28], [29]; 2) point-based [30], [31], [32], [33], i.e., using points to present a surface; 3) mesh-based [34], [35], [36], [37], [38], [39], [40], [41], [42], [43], [44], i.e., describing an object with vertices, edges, and faces; and 4) implicit neural representation [45], [46], [47], [48], [49], [50], [51], [52], [53], [54], [55], [56], i.e., representing a shape as a neural network that takes any (x, y, z) coordinate as input and maps it to an occupancy or signed distance of the shape at that coordinate or modeling radiance or appearance properties of an object such as NeRF-based approaches [57], [58], [59], [60], [61], [62], [63], [64], [65], [66], [67], [68].

In this survey, we present a comprehensive overview of these state-of-the-art DL-based approaches to 3-D surface reconstruction. Our main goal is to provide method researchers with a guide to current work and applied researchers with a toolbox for their domain challenges. Toward this end, we first provide a broad introduction to input data formats (see Section II), acquisition technologies (see Section III), and widely used benchmarking datasets (see Section IV). Section V covers evaluation metrics enabling to quantitatively judge the reconstructive performance of a method, independent of being classical or learning-based. The main part of this survey (see Section VI) highlights DL methods to reconstruct 3-D surfaces using volumetric, point- and mesh-based, and implicit neural representations. We assume that the reader has a general grasp of neural networks and DL concepts to thoroughly follow the content. Discussion, current trends, and challenges are highlighted in Section VII. Finally, Section VIII summarizes and concludes this survey.

II. INPUT DATA

Various types of data representations can be used as input for the 3-D surface reconstruction task. Conventional representations of 3-D inputs can be divided into Euclidean and non-Euclidean data. Examples of non-Euclidean data representations are point clouds or meshes, while Euclidean data representations can be volumetric, RGB-D data, or multiview images.

Point clouds are currently the most common format of raw 3-D sensor data. With the improvement of scanning devices, leading to enhanced capabilities for capturing the

surrounding 3-D environment in various applications and representing it with points, point clouds are becoming increasingly important and available. Thus, processing this type of representation using neural networks and DL techniques has attracted considerable attention. From a mathematical point of view, point clouds comprise an irregular data structure in the form of an unordered set of points. Each point on a 3-D surface of an object can basically be defined by a vector of its (x, y, z) coordinates, which can be inferred by various 3-D data acquisition techniques. Hence, the size of the representation matrix of a 3-D object is initially $N \times 3$ for N points. The matrix may also contain different properties including color, transparency, surface normals, and other scanner information. However, pure point clouds do not include the interconnections between vertices. Since a point cloud is a set, its elements are orderless, a characteristic that causes many challenges for surface reconstruction methods. Point clouds can be easily converted to/extracted from other data representations, such as voxels, depth maps, or meshes, and vice versa. Furthermore, they can be extracted from depth images by projecting the depth value of each pixel into 3-D space.

Meshes are another highly popular type of representation for 3-D objects providing detailed and connected geometries in an efficient way. They are irregular data embedding in continuous space. Their basic components are vertices, edges, i.e., pairs of vertices, and (triangular) faces, i.e., n-tuples of edges, forming an undirected graph.

In volumetric representations, the basic element is a voxel. A voxel in a 3-D grid is a cuboid equivalent to a pixel in 2-D space. The 3-D grid, regardless of being sparse or dense, can be fed to a neural network as the input.

An RGB-D image is a combination of an RGB image and a depth image. It not only has RGB information for each pixel but also includes depth information.

Multiview images are a collection of (single-view) images taken from different angles of an object. By putting these images together, 3-D information can be partly retrieved.

On the other hand, 2-D data, such as single-view RGB images, can also be considered the input to a network for surface reconstruction individually, in which the method is called single-view reconstruction (SVR) [69], [70], [71] or in conjunction with another 3-D input mentioned earlier.

III. DATA ACQUISITION

As explained in Section II, point clouds are the most common format of raw 3-D sensor data. 3-D point cloud data are acquired through sensing technologies that measure distance [i.e., 3-D laser scanning also known as light detection and ranging (LiDAR)] or generated with stereo- and multiview image-derived systems that can be based on red, green, blue-depth (RGB-D) cameras, stereo cameras, and multiple synthetic aperture radar (SAR) image pairs [16], [72]. High-quality 3-D point clouds can capture the 3-D surface geometries of target objects (e.g., physical features that occupy the Earth's surface and ocean bottom) with a spatial accuracy up to the millimeter level and a point density of a few thousand points per square meter (pts/m^2).

A. 3-D Laser Scanning (LiDAR)

LiDAR is a remote sensing (RS) active technology that uses light in the form of a pulsed laser to measure the distance between the sensor and the object under study [73]. By measuring the time that emitted pulses take to travel to a target, LiDAR derives 3-D representations of objects. LiDAR can also operate at different wavelengths (i.e., multispectral LiDAR [74], [75]) to discriminate the different spectral reflectance of land-cover classes [76], [77].

Depending on the platform on which the LiDAR sensor is mounted, a 3-D laser scanner is classified as a terrestrial laser scanner (TLS or ground LiDAR), airborne laser scanner (ALS), mobile laser scanner (MLS), and unmanned laser scanner (ULS) [16], [72].

A TLS uses ground-based RS systems (e.g., tripods) to cover middle- or close-range areas with scans performed in all directions, including upward [78]. Once scans of a single zone are completed, the tripod is moved to another location to scan from another angle or capture data from a new area. As TLS systems are static during the acquisition process, they reach the highest point cloud density and can produce high-quality 3-D models of the interiors of buildings and heritage sites.

Nevertheless, TLS systems cannot always be used, especially for scanning restricted locations that are not safe or accessible for teams (e.g., areas of dense vegetation and unsafe building sites). In these cases, LiDAR sensors can be mounted on airborne platforms. ALS systems are also used to acquire point cloud data over large areas (e.g., for 3-D building reconstruction [79]).

When target regions are directly accessible, their structures and objects can be reconstructed from data acquired by MLS systems, i.e., LiDAR sensors mounted on moving vehicles (e.g., to derive high-resolution 3-D city models [80]).

Since drones and other unmanned vehicles have become cheaper and autonomous navigation more reliable [81], ALS and MLS are often operated as ULS systems. Their platforms are compact and lightweight, which enables them to be exploited as first responders for disaster management. ULS systems can make a first scan of the terrain to track movements and changes, and deliver 3-D mapping of the most affected locations [82], [83].

B. Photogrammetry

While LiDAR performs a direct measurement of the target object, i.e., by physically hitting a feature with light and measuring the reflection, approaches based on photogrammetry or computer vision theory [84] use a set of overlapping images taken from different locations to identify isolated points within a target. This includes not only airborne photogrammetry but also satellite stereo

systems, which can map larger regions quickly. Image-based reconstruction algorithms can estimate the relative locations of these points and eventually convert the overlapping images into a 3-D point cloud. For instance, the structure from motion (SfM) algorithms [85] can process multiview images simultaneously through estimating camera positions and orientations automatically, while dense matching and multiview stereo (MVS) algorithms [86] can generate a large volume of point clouds (e.g., large-scale scenarios and crowded environments).

C. RGB-D Camera

Similar to LiDAR, RGB-D cameras measure the distance between the sensor and the objects. Depth information of each RGB pixel of the image is retrieved via a depth sensor. An RGB-D camera generates a colored point cloud by mapping RGB images with depth information (i.e., images include the (x, y, z) spatial coordinates and RGB colors). In this case, the point cloud is not the direct result of RGB-D scanning [87], [88] since the camera generates pixelwise depth data rather than unstructured points. RGB-D cameras are generally cheaper than LiDAR systems and are mostly used in indoor environments for close-range applications [89].

Structured light and Time of Flight (ToF) [90], which are active imaging systems, serve as depth cameras and calculate the distance from the sensor to an object, consequently providing 3-D information. The depth of an object can be determined using ToF sensors by measuring the duration of light travel from the sensor to the object and back. By determining the ToF of light, these sensors can calculate the object's distance and create a detailed depth map, which can be directly used or easily converted to a point cloud for instance. Structured light sensors employ the deformation of a projected pattern to determine the distance. By emitting a known light pattern onto a scene and examining how the pattern changes as it interacts with objects in the scene, these sensors are able to accurately measure the depth information of the objects. Structured light technology-based 3-D scanners are comparatively more affordable, being lighter in weight than their laser-based counterparts as well. Due to their higher degree of sensitivity to lighting conditions, they may not operate well in outdoor environments or in challenging conditions such as dusty rooms. For black or glossy surfaces, a specific spray should be applied before 3-D scanning.

D. SAR Point Cloud

SAR is an active RS system that can operate day and night and can penetrate clouds and smoke. Interferometric SAR (InSAR) extends the principle of SAR to the 3-D domain [91] by taking advantage of the physical properties of microwaves [92]. An InSAR system compares the phase of multiple SAR image pairs acquired from slightly different viewing angles to generate InSAR-based point clouds. The SAR tomography (TomoSAR) and persistent scatterer interferometry (PSI) are two major techniques that generate point clouds with InSAR [16]. They are used to monitor terrain changes (e.g., surface deformations and human-made structures [93]).

E. Videogrammetry

3-D point clouds can also be reconstructed using video frames (i.e., the input data are video streams instead of a collection of images). This approach is referred to as videogrammetry [94] and is based on the principles of photogrammetry. It can reconstruct point clouds from the frames of a video since their information is sequentially interconnected. Videogrammetry approaches provide a valuable alternative to camera images. They can be semi-automatic since the search for target points in different images can be achieved by measuring or tracking features of interest between consecutive video frames. However, the reconstruction needs to be coupled with effective frame selection algorithms (e.g., video frames are selected based on the surveyed geometry) and robust 3-D processing methodologies [95].

IV. DATASETS

DL approaches are data-demanding; thus, they require large amounts of data with high-quality 3-D shapes and ground truths. Recent developments in scanning and sensing technologies have led to the collection of various widely used and openly accessible benchmarking datasets. These datasets are used to train and evaluate the performance of DL methods for different tasks, including 3-D reconstruction. In this section, we summarize some of the most popular datasets, which can be used by different 3-D DL approaches, with a focus on 3-D reconstruction. Table 1 offers a comparative overview of these datasets.

1) ShapeNet [96] is a richly annotated, large-scale synthetic dataset of 3-D shapes represented by 3-D computer-aided design (CAD) models of objects, providing roughly 3 000 000 shapes. This dataset has been used for computer graphics and vision purposes. The full ShapeNet dataset is not yet publicly available. It consists of several subsets, including ShapeNetCore and ShapeNetSem. ShapeNetCore contains a single clean 3-D shape that covers 55 common object categories with about 51 300 unique 3-D shapes. ShapeNetSem is a smaller, more densely annotated subset, containing 12 000 shapes of a broader set of 270 categories. For each shape in ShapeNet, annotations such as its geometry, texture, parts, symmetry planes, voxelization, screenshot, category, alignment, and size are available. The final representation of an object in this dataset can be a 3-D mesh. The 3-D shapes are stored in the Wavefront object file format (`.obj`), which describes the surface geometry of a 3-D shape and includes vertices and faces, along with material template library (`.mtl`) files used to store material definitions. An `.mtl` file is a companion file

Table 1 Comparison of Benchmark Datasets

Name	Count/Size	Dataset Type	Representation	Scene Type	Source	DL Tasks
ShapeNetCore	51,300 3D models from 55 object categories	Synthetic	Mesh	Indoor and outdoor objects	CAD model	Shape recognition, reconstruction, retrieval
ShapeNetSem	12,000 3D models of 270 object categories	Synthetic	Mesh	Indoor and outdoor objects	CAD model	Shape recognition, reconstruction, retrieval
PartNet	573,585 part instances of 26,671 3D ShapeNet models in 24 object categories	Synthetic	Mesh and point cloud	Indoor object parts	CAD model	Part-level understanding
ModelNet	127,915 3D models with 662 object categories	Synthetic	Mesh	Indoor and outdoor objects	CAD model	Recognition, reconstruction, generation, and completion
KITTI	Around 49,000 frames from 5 categories	Real-world	Image and point cloud	Outdoor	RGB and LiDAR	Stereo, optical flow, visual odometry, SLAM, 3D object detection, and object tracking
Semantic KITTI	23,201/20,351 scans with 4549 points from 28 classes	Real-world	Point cloud	Outdoor	LiDAR (MLS)	Semantic segmentation and scene completion
ScanNet	2,5 million frames from 1500 RGB-D scans	Real-world	Image and mesh	Indoor	RGB-D Sensor	Object classification, voxel labeling, model retrieval, and reconstruction
Matterport3D	10,800 views from 90 scenes	Real-world	Image and mesh	Indoor	RGB-D Sensor	Scene understanding, normal prediction, classification, semantic segmentation, and reconstruction
NYU depth v2	1449 RGB-D images consisting 464 diverse scenes across 26 scene classes	Real-world	Image	Indoor	RGB-D	Segmentation
Sun3D	415 sequences captured for 254 different spaces in 41 different buildings	Real-world	Image and point cloud	Indoor	RGB-D sensor	Scene understanding, reconstruction, and segmentation
Sun RGB-D	10,335 RGB-D images from 47 scene categories consisting about 800 object categories	Real-world	Image	Indoor	RGB-D sensor	Scene understanding, semantic segmentation, object detection, orientation, and classification
Sydney urban objects	631 scans from 26 object categories	Real-world	Point cloud	Outdoor	LiDAR	Classification and recognition
ABC	1 million 3D models	Synthetic	Mesh	Indoor	CAD model	Feature detection, shape reconstruction and surface normal estimation
Semantic3D	4 billion points in 8 class labels	Real-world	Point cloud	Outdoor	LiDAR (TLS)	Classification and semantic segmentation
H3D	Around 73 million points and 3,5 million faces in 11 classes	Real-world	Point cloud and mesh	Outdoor	LiDAR	Semantic segmentation

Table 1 *(Continued.)* Comparison of Benchmark Datasets

3D-Front	18,968 rooms with 13,151 furniture objects from 31 scene categories	Synthetic	Mesh	Indoor	CAD model	3D scene understanding, reconstruction, and segmentation
3D-FUTURE	20,240 images of 5,000 different rooms	Synthetic	Mesh	Indoor	CAD model	2D instance segmentation, 3D object pose estimation, image-based 3D shape retrieval, 3D reconstruction from a single image, and texture recovery for 3D shape
SensatUrban	4 billion points in 13 semantic class labels	Real-world	Image and point cloud	Outdoor	UAV Photogrammetry	Urban-scale point cloud understanding

for one or more `.obj` files, which describes some surface appearance properties.

2) PartNet [97] is a dataset of 3-D objects, built on top of ShapeNet with fine-grained, hierarchical, and instance-level 3-D part annotations. The dataset comprises 573 585 part instances of 26 671 ShapeNet 3-D shapes in 24 indoor object categories in an attempt to enable part-level understanding of 3-D objects.
3) ModelNet [98] is a large-scale CAD model synthetic dataset. It includes a comprehensive and clean collection of 127 915 CAD models with 662 object categories and consists of two subsets, ModelNet10 and ModelNet40 with ten and 40 classes, respectively. ModelNet10 has also been annotated with the orientation of the CAD models, which are given in the Geomview object file format (`.off`). The final representation of this dataset can be a mesh.
4) KITTI [99], [100] is a real-world urban scene dataset composed of images and point clouds. The dataset was acquired by the autonomous driving platform Annieway while driving around the city of Karlsruhe. Evaluation benchmarks were developed for several computer vision and robotic tasks, such as stereo, optical flow, visual odometry, SLAM, 3-D object detection, and 3-D object tracking. Semantic KITTI [101], which is based on KITTI, provides pointwise annotations for semantic segmentation and semantic scene completion purposes. The dataset comprises 28 classes including classes for nonmoving and moving objects.
5) ScanNet [102] is a 3-D reconstruction dataset of indoor scenes consisting of 2.5 million frames (views) derived from more than 1500 RGB-D scans. 3-D camera poses, surface reconstructions, and instance-level semantic segmentations are also provided. All scans are reconstructed into 3-D mesh models. The data are stored in polygon file format (`.ply`).
6) Matterport3D [103] is another dataset facilitating RGB-D scene understanding. It captures 10 800 panoramic views from 194 400 RGB-D images of 90 building-scale scenes. The dataset is annotated with surface reconstructions as textured meshes, camera poses, and 2-D/3-D semantic segmentations.
7) NYU depth v2 [104] introduced an annotated dataset of 1449 RGB and depth images, consisting of 464 diverse indoor scenes. These images were acquired by RGB and depth cameras from Microsoft Kinect.
8) Sun3D [105] is a real-world large-scale dataset of RGB-D frames with semantic object segmentations and camera pose used for scene understanding. It consists of 415 sequences captured for 254 different indoor spaces in 41 different buildings.
9) SUN RGB-D [106] is a dataset containing over 10 000 RGB-D images from NYU depth v2 [104], Berkeley B3DO [107], and SUN3D [105] datasets. These images are annotated with 2-D segmentations (146 617 2-D polygons), 3-D object boxes (64 595 3-D bounding boxes), 3-D room layout, 3-D object orientation, and scene category for each image.
10) The Sydney Urban Objects dataset [108] is a point cloud dataset that contains 631 scans of 26 different object classes, including vehicles, pedestrians, trees, and signs taken in the city of Sydney. Each object's information is available in three file formats: ASCII CSV format (`.csv`), binary-packed CSV (`.bin`), and meta information files (`.meta`).
11) The ABC dataset [109] is a CAD model dataset with one million 3-D models. Koch et al. [109] offered a pipeline that is able to convert these CAD models into other representations in order to be processable by DL techniques. These models are provided in `.obj` and 3-D systems' stereolithography CAD file format (`.stl`).
12) Semantic3D.net [110] is a large labeled 3-D point cloud dataset of natural scenes with over four billion points in eight class labels. These dense point clouds, which were recorded by TLSs, depict urban and rural outdoor terrestrial scenes.
13) H3D [111] is a high-resolution real-world dataset containing both point clouds (H3D(PC)) and meshes (H3D(Mesh)) of airborne LiDAR data and can be used for semantic segmentation in geospatial applications.

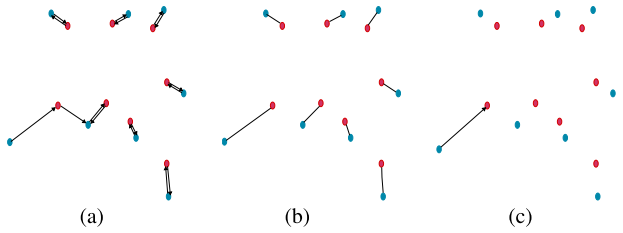


Fig. 2. *Visualization of (a) CD, (b) EMD, and (c) HD metrics. Red dots and blue dots belong to two different point sets, and each of these metrics measures the distance between these two sets in a unique way.*

The point clouds are classified into 11 classes, and labeled 3-D textured meshes can be derived from them.

14) 3-D Furnished Rooms with layOuts and semaNTics (3D-Front) [112] is a synthetic dataset of indoor CAD model scenes, containing 18 968 rooms with 3-D objects. The individual objects are taken from 3D-FUTURE [113]. The CAD models are stored in `.obj` and `.mtl` file formats.
15) 3-D Furniture shape with TextURE (3D-FUTURE) [113] is a repository of 3-D furniture shapes in the household scenario enriched with 3-D and 2-D annotations. It includes 20 240 synthetic images of 5000 different rooms. Stylistic and texture details of individual objects are provided. The 3-D models are stored in `.obj` file format.
16) SensatUrban [114] is a dataset for urban-scale point cloud understanding. It covers 7.6 km^2 of urban areas in Birmingham, Cambridge, and York cities. The point clouds are obtained from high-resolution aerial images, which are captured by the UAV mapping system.
17) Stanford 3-D Scanning Repository [115] is a surface reconstruction repository containing some famous 3-D models, such as the Stanford bunny, happy Buddha, dragon, and armadillo in `.ply` format. These 3-D models and some others also exist in the Large Geometric Models Archive [116].

V. EVALUATION METRICS

Evaluation metrics are used to assess the performance of DL models [1], [2], [3]. Various metrics have been proposed for testing deep geometric learning methods. Some of the common distance metrics used for surface reconstruction methods are Chamfer distance (CD), earth mover's distance (EMD), and Hausdorff distance (HD), which all measure the discrepancy between two sets, as illustrated in Fig. 2. Another common metric for evaluating 3-D reconstruction solutions is the Intersection over Union (IoU). Furthermore, the formulas in this section denote false positives, false negatives, true positives, and true negatives as FPs, FNs, TPs, and TNs, respectively.

1) The CD [30] measures the distance between two different surfaces or sets of points by first calculating the distances between predicted points and their ground-truth nearest neighbors and then averaging all of these distances. The calculated value represents the dissimilarity between predicted output and ground truth. The lower the value, the better the result. Let S_1 and S_2 be two point clouds that represent the predicted and ground-truth shapes, and x and y be two points that belong to these point clouds, respectively. Then, the CD is defined as

$$d_{\text{CD}}(S_1, S_2) = \sum_{x \in S_1} \min_{y \in S_2} ||x - y||_2^2 + \sum_{y \in S_2} \min_{x \in S_1} ||x - y||_2^2. \quad (1)$$

2) The EMD, also known as the Wasserstein distance in mathematics and optimization theory) [30], [117], [118], is based on solving an optimization problem, called the transportation problem. The transportation problem attempts to find the least-expensive flow of goods from suppliers to consumers while satisfying the consumers' demand. For the calculation of the EMD of two point sets, each point in one set should be assigned to a unique point in the other set to fulfill optimal assignment. EMD uses bijection between the points that minimize the total sum of the pairwise distances. Consider $S_1 \subseteq R^3$ and $S_2 \subseteq R^3$ to be two point sets of equal size, representing the predicted and ground-truth shapes, respectively. The EMD [30] is defined as

$$d_{\text{EMD}}(S_1, S_2) = \min_{\phi: S_1 \to S_2} \sum_{x \in S_1} ||x - \phi(x)||_2 \quad (2)$$

where $\phi : S_1 \to S_2$ is a bijection.

3) The HD considers the farthest and largest dissimilarity between predicted output and ground truth. A point in one set that has the worst mismatch and maximum distance from its nearest point in the other set determines the HD

$$d_{\text{HD}}(S_1, S_2) = \max \left\{ \max_{x_i \in S_1} \min_{y_j \in S_2} ||x_i - y_j|| \; \max_{y_j \in S_2} \min_{x_i \in S_1} ||x_i - y_j|| \right\}. \quad (3)$$

The metric is, however, not very robust toward outliers.

4) The IoU, also known as the Jaccard Index, is often used as a quality measure in object detection and semantic segmentation. As illustrated in Fig. 3, it is defined as the overlap between the prediction and the ground truth, divided by their union. The lower the IoU, the worse the prediction result.

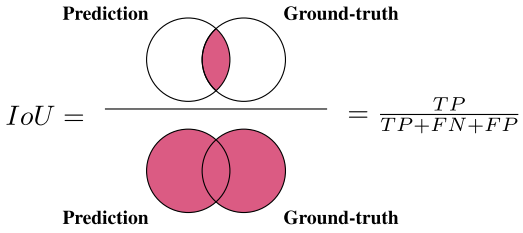


Fig. 3. *Visual intuition of the IoU metric.*

IoU can also be easily utilized for evaluating voxel-based representations and specifying the overlap between a reconstructed 3-D voxel and its voxelized ground truth. For volumetric approaches, IoU can be formulated as [20]

$$\text{IoU} = \frac{\sum_{i,j,k}\left[I\left(p_{(i,j,k)} > t\right) I\left(y_{(i,j,k)}\right)\right]}{\sum_{i,j,k}\left[I\left(I\left(p_{(i,j,k)} > t\right) + I\left(y_{(i,j,k)}\right)\right)\right]} \quad (4)$$

where $I(.)$ is an indicator function, $p(i,j,k)$ is the predicted voxel occupancy probability, t is a voxelization threshold, and $y(i,j,k)$ is the ground-truth occupancy probability.

5) In classification problems, precision is the number of predictions correctly assigned to one label, i.e., true positives, divided by the number of all predictions assigned to that label, including those identified incorrectly, i.e., false positives (see Fig. 4)

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}. \quad (5)$$

The average precision (AP) is computed by averaging all precision values of all positively labeled samples [98]. The mean AP (mAP) is the average of AP calculated over all classes. For point clouds, precision is calculated as the percentage of predicted points that are close to the ground-truth surface, i.e., with a distance less than a specific threshold [119].

6) Recall or sensitivity denotes the ratio between the number of predictions correctly assigned to one class (TP) and the actual number of elements in that class, including those that are incorrectly assigned to the other label (FN) (see Fig. 4). It is a measure of how well a DL model can find all labels of one class

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \quad (6)$$

For point clouds, recall is calculated as the percentage of points on the ground truth, which are close to the predicted surface, i.e., having a distance less than a specific threshold [119].

7) The F_1 score, also known as balanced F-score, F-measure, or dice similarity coefficient (DSC), is the harmonic mean of precision and recall. The higher the value, the better the result

$$F_1\text{score} = 2 * \frac{(\text{Recall} * \text{Precision})}{(\text{Recall} + \text{Precision})}. \quad (7)$$

For point clouds, precision and recall can be calculated by checking the percentage of points in one point cloud, for instance, the predicted point cloud or the ground truth, which can find a neighbor from the other point cloud within a threshold [38]. Intuitively, the F-score can be interpreted as the percentage of points that were reconstructed correctly [119].

8) In classification problems, the accuracy (Acc) is the ratio between correct predictions and all predictions, i.e., it shows how much of the data is labeled correctly

$$\text{Accuracy} = \frac{(\text{TP} + \text{TN})}{(\text{TP} + \text{FP} + \text{FN} + \text{TN})}. \quad (8)$$

However, it is not an appropriate metric for imbalanced datasets as it does not take into account the distribution skew [120].

9) Normal consistency (NC) [45] is defined as the mean absolute dot product of the surface normal of each point, i.e., a perpendicular vector to the surface at the given point, in one mesh, and the surface normals of its nearest neighbors in the other mesh

$$\begin{aligned}\text{NC}\left(\hat{M}, M\right) &= \text{Normal Consistency}\left(\hat{M}, M\right) \\ &= \frac{1}{2|\partial\hat{M}|}\int_{\partial\hat{M}} |\langle n(p), n(\pi_2(p))\rangle| dp \\ &+ \frac{1}{2|\partial M|}\int_{\partial M} |\langle n(\pi_1(q)), n(q)\rangle| dq\end{aligned} \quad (9)$$

where $\partial\hat{M}$ and ∂M are predicted and ground-truth mesh surfaces, $n(p)$ and $n(q)$ are unit normal vectors on these mesh surfaces, respectively, $\pi_2(p)$ and $\pi_1(q)$ indicate the projections of p and q on the aforementioned surface meshes, respectively, and $\langle .,. \rangle$ implies the inner product. The higher the NC, the better the result.

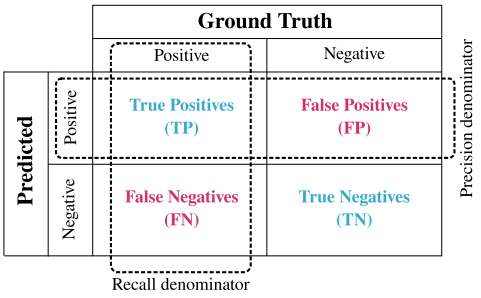


Fig. 4. *Confusion matrix for binary classification.*

10) The Jensen–Shannon divergence (JSD) [31] measures the similarity between marginal point distributions. It is mainly based on the Kullback–Leibler (KL) divergence [121]. Considering two point clouds and a voxel grid that discretizes 3-D space, the number of points within each voxel from the predicted point set P and the ground-truth point set G is counted. The JSD between the obtained empirical distributions (P_P, P_G) is calculated as

$$\text{JSD}\left(P_P||P_G\right) = \frac{1}{2}D_{\text{KL}}\left(P_P||M\right) + \frac{1}{2}D_{\text{KL}}\left(P_G||M\right) \tag{10}$$

where $M = (1/2)(P_P + P_G)$.

11) Coverage [31] quantifies the fraction of points in the ground-truth set S_2, which are matched to points in the predicted set S_1. A match happens when the nearest neighbor in the ground-truth set is found for each point in the predicted set

$$\text{Coverage}\left(S_1, S_2\right) = \arg\min_{Y \in S_2} \frac{D\left(X, Y\right) |X \in S_1|}{|S_2|} \tag{11}$$

where $D(.,.)$ or "nearness" is measured using distance metrics, such as CD or EMD. High coverage indicates that most of the points in S_2 are roughly present within S_1. However, this does not assess the quality of the predicted set. Achieving perfect coverage is possible, despite large distances between the predicted point set and the ground-truth set [33].

12) Minimum matching distance (MMD) [31], [33] is a complement to the coverage metric. It measures the distance between every point in the ground-truth set S_2 and its nearest neighbor in the predicted set S_1 and averages these distances in order to evaluate the quality of the predicted set

$$\text{MMD}\left(S_1, S_2\right) = \frac{1}{|S_2|} \sum_{Y \in S_2} \min_{X \in S_1} D\left(X, Y\right) \tag{12}$$

where $D(.,.)$ is measured using distance metrics such as CD or EMD.

13) Light field descriptor (LFD) [122] measures visual similarity between 3-D shapes. In short, LFD assumes that a 3-D object can be represented as a number of 2-D views; therefore, if two 3-D models are similar, they also look alike from all views. A light field, which is used in image-based rendering, is defined as a 5-D function that represents the radiance at a given 3-D point along a given direction. To extract LFD for a 3-D model, a set of image renderings (silhouettes) are obtained from different angles. These rendered images are acquired using cameras located on the vertices of a fixed regular dodecahedron, i.e., 20 vertices, which surrounds the 3-D model. Each of these silhouettes is then encoded both by a region shape descriptor (Zernike moments descriptor) and a contour shape descriptor (Fourier descriptor) for similarity comparisons. A visual representation can be found in Fig. 5. LFD is a good visual similarity metric for 3-D surfaces; however, by rendering merely the silhouette of the shape without lighting, LFD can only observe the condition of this shape on the edge of the silhouette [49]. D_A, which is the dissimilarity between two 3-D models, is calculated as

$$D_A\left(L_1, L_2\right) = \min_i \sum_{k=1}^{10} d\left(I_{1k}, I_{2k}\right), \quad i = 1, \ldots, 60 \tag{13}$$

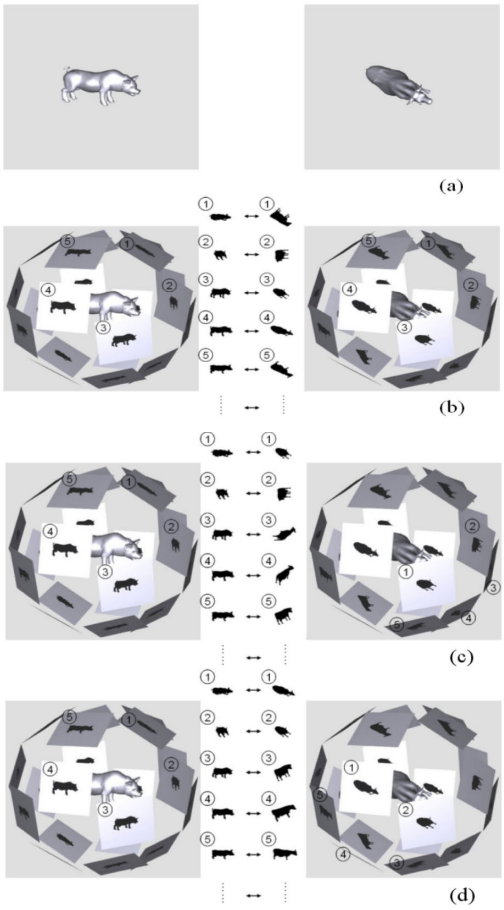


Fig. 5. *(a) Comparison of LFDs between two 3-D models: a pig and a cow. First, rendered images are extracted for both 3-D models. Then, as illustrated in (b), all 2-D images from the same views are compared, and a similarity value for this camera angle is obtained. Next, a different mapping between rendered images of the two 3-D models is chosen, and thus, another similarity value is extracted, as illustrated in (c). Eventually, the rotation of camera positions with the best similarity is found, as shown in (d). The similarity between the two 3-D models is attained by summing up the similarities from all the corresponding images [122].*

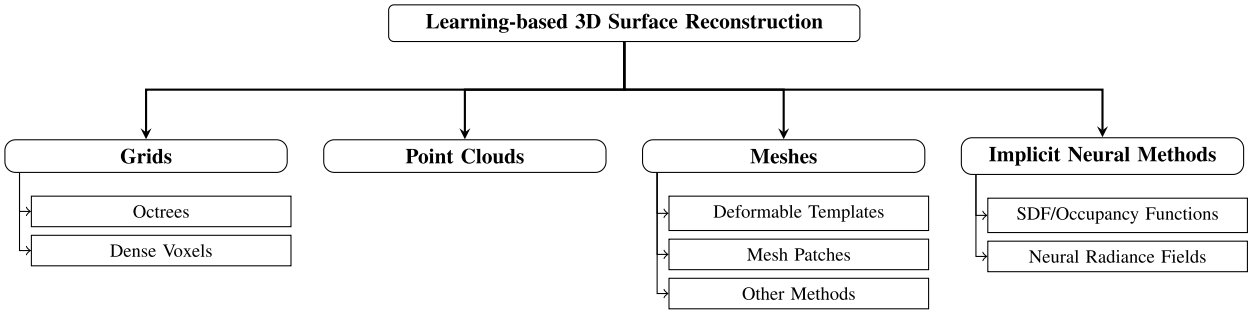


Fig. 6. *Taxonomy of learning-based reconstruction approaches based on 3-D shape representation.*

where i indicates different rotations between camera positions for two 3-D models, and I_1k and I_2k are corresponding images for the ith rotation. The dissimilarity between two images is denoted by d.

VI. DL-BASED 3-D SURFACE RECONSTRUCTION

DL-based 3-D surface reconstruction approaches can be broadly classified into four main categories according to their representation, as illustrated in Fig. 6:

1) Volumetric representations define a surface via small cuboids, either a dense 3-D voxel grid [20], [21], [22], [23], [24], [25] or an octree [26], [27], [28], [29]. Dense voxels are the 3-D analog of a pixel in 2-D space, i.e., a cubical element in a regularly spaced 3-D grid. Therein, octrees are obtained by recursively splitting 3-D space into octants, i.e., eight equally sized cells. In this data structure, only cells containing information by being close to the surface boundary are subdivided. Neighboring cells that have the same value do not need to be subdivided, and all of these areas can be represented by a single large octree cell. In order to achieve finer details, the space can be further partitioned into smaller octree cells, which is the main difference between a regular voxel grid and an octree.
2) Point-based representations utilize the constituting surface points to mark a shape [30], [31], [32], [33]. The entire surface is described through an unordered set of (x, y, z) coordinates.
3) Mesh-based representations describe an object through vertices, edges, and faces [34], [35], [36], [37], [38], [39], [40], [41], [42], [43], [44]. Existing approaches can be mainly divided into three categories.
   a) Patch-based approaches attempt to reconstruct the final shape by learning a group of mappings from 2-D squares to 3-D patches and putting together these small patches.
   b) Deformable template-based approaches deform the vertices of a template mesh with predefined interconnections and predict the final shape based on it.
   c) Other mesh generation methods are so unique, yet singular, that they are sorted into a catch-all category.
4) Implicit neural representations describe a shape as a neural network that takes any (x, y, z) coordinate as input and maps it to occupancy or signed distance value [45], [46], [47], [48], [49], [50], [51], [52], [53], [54], [55], [56] or model radiance or appearance properties of an object such as NeRF-based approaches [57], [58], [59], [60], [61], [62], [63], [64], [65], [66], [67], [68].

Accordingly, we summarize and discuss the existing literature for DL-based 3-D surface reconstruction methods based on these categories in Sections VI-A–VI-D. Furthermore, we depict the architecture of different approaches with a unique color scheme in these sections. In the figures, data units are represented in red, trainable units in blue, and computing units in orange.

A. Volumetric Representations

Volumetric approaches in neural networks for 3-D surface reconstructions rely on describing the object through a grid. By extending the concept of 2-D convolutions to 3-D, a grid can be easily processed using learning-based approaches, such as neural networks.

Volumetric methods characterize 3-D object data using: 1) a regular 3-D voxel grid, i.e., dense voxels, and/or 2) an octree, i.e., sparse voxels.

Analogously to the concept of a pixel in the 2-D world, a voxel is a cubical element in a regularly spaced 3-D grid. An octree can be built by recursively subdividing the space into octants until a predefined maximum depth is reached. Additional information can be stored in cubic cells (both in dense and sparse voxels) to help reconstruct surfaces as follows.

1) Signed distance functions (SDFs) express the distance between the center of each voxel and the closest point on the surface of an object. They can be stored in a cuboid by calculating distance functions (DFs) [25], [123]. SDFs, a variation of DFs, purely calculate the signed distance value for each cell. Truncated SDFs (TSDFs) [124] go beyond the SDF definition by specifying a truncation threshold for SDF values stored in

cuboids, i.e., assigning a fixed value to voxels that are not near enough to the surface and their signed distance values exceed the defined threshold.

2) Occupancy or indicator functions indicate whether a cuboid is occupied by the surface of an object or not.

Learning voxel-based SDF representations is usually rather complicated compared to occupancy representations since dealing with DFs in 3-D space is more difficult than simply classifying a voxel as occupied or unoccupied [45]. However, voxel-based SDF approaches provide the advantage of generating smoother surfaces compared to occupancy grid-based approaches. A general disadvantage of voxel-based methods is their resolution limitation by the underlying 3-D grid. Mesh extraction approaches, such as the classical Marching Cubes (MC) algorithm [125], can be used to infer a mesh from the final output of these methods.

1) Dense Voxels: The majority of approaches with dense voxel-based representation voxelize the 3-D space in order to apply 3-D convolutional neural networks (CNNs) on a grid directly. In this section, we first present pioneer studies that applied CNNs to a 3-D representation, i.e., dense voxels, for shape classification and then introduce 3-D surface reconstruction and shape completion approaches that use dense voxels.

a) Volumetric CNNs for 3-D shape classification: Several studies have focused on solving shape classification and recognition tasks using dense voxels [21], [23], [98], [126], [127], [128], [129]. One of the pioneers in building DL models in 3-D world is 3-D ShapeNets, as proposed by Wu et al. [98]. They were among the first authors to show the application of CNNs to a 3-D representation. The introduced architecture uses a convolutional deep belief network for representing a 3-D shape as a probabilistic distribution of binary variables on a 3-D voxel grid. 3-D ShapeNet is able to conduct several tasks, from shape recognition to reconstruction and completion, as well as next-best-view prediction. The DL model takes a single-view depth map of the physical object as input and converts it into a volumetric representation. The occupancy status of each cell is specified by classifying it as either free space, unknown space, or observed surface. Next, a deep belief network is trained on this grid of size 30^3. In terms of accuracy, precision, and recall metrics, 3-D ShapeNets outperform several baseline methods for 3-D shape classification and retrieval, such as the LFD approach [122] and the spherical harmonic descriptor (SPH) [130], even though it utilizes a mesh at lower resolution. It was further shown that the DL model is able to automatically learn general 3-D features.

Maturana and Scherer [126] introduced VoxNet that voxelizes input point cloud data and processes the grid with a 3-D-CNN for object recognition tasks. The authors utilized a volumetric grid for representing the estimated spatial occupancy and a 3-D-CNN for extracting features and predicting class labels directly from the occupancy grid of size 32^3. Each point in the input point cloud is mapped to discrete volume coordinates. The resulting voxel volumes are fed to the proposed shallow neural network. VoxNet has fewer parameters compared to 3-D ShapeNets [98], i.e., less than one million versus over 12.4 million parameters, while achieving 8% and 6% higher average accuracy for ModelNet10 and ModelNet40 datasets, respectively. However, in both these methods, the memory and computational costs increase cubically with respect to the input resolution.

ORION [127], which is based on VoxNet [126], studies the importance of object orientation in 3-D object recognition results. Unlike VoxNet and 3-D ShapeNets [98], which augment training data with rotations of the objects to achieve rotational invariance of the network, ORION seeks to predict object orientation. The proposed network uses 3-D convolutional networks for 3-D recognition and adds an auxiliary orientation loss for better classification performance. By forcing the network to predict object orientation in addition to class labels during training, more accurate classification results can be achieved at test time. The ORION network is shallower than the proposed method by Brock et al. [21] that we discuss further down this survey, leading to fewer trainable parameters.

Some studies utilize multiview CNNs for analyzing a 3-D shape. Multiview CNNs work in three steps: 1) rendering a 3-D shape as a collection of images from different viewpoints; 2) inferring features for each viewpoint; 3) fusing these features across various views. In order to minimize the performance gap between multiview CNNs and volumetric CNNs, Qi et al. [128] suggested two new network architectures of volumetric CNNs. One architecture focuses on local regions, while the other uses anisotropic probing kernels for convolving a 3-D cube, then projecting 3-D volumes to a 2-D image, and afterward applying image-based CNNs for classification. The proposed CNNs surpass volumetric CNN-based methods, such as 3-D ShapeNets [98] and VoxNet [126]. Moreover, their classification accuracy competes with some multiview-based methods, such as MVCNN [131], LFD approach [122], and SPH [130], given the same 3-D resolution of 30^3.

b) 3-D surface reconstruction and shape completion using volumetric representation: In this section, we review the studies that leverage dense voxel representations for 3-D surface reconstruction [20], [21], [22], [23] and 3-D shape completion [24], [25]. Choy et al. [20] introduced a framework, 3-D recurrent reconstruction network (3D-R2N2), for both single- and multiview 3-D reconstructions. This method takes one or more RGB images of an object from arbitrary viewpoints as input and outputs a 3-D occupancy grid. The proposed network is composed of three main modules, as shown in Fig. 7: 1) a 2D-CNN, which encodes the input into a low-dimensional feature vector; 2) a 3-D convolutional long short-term memory (LSTM) [132], in which the 3D-LSTM units keep their previous cell states or update them, whenever there are more observations, i.e., multiview images, available; 3) a 3-D

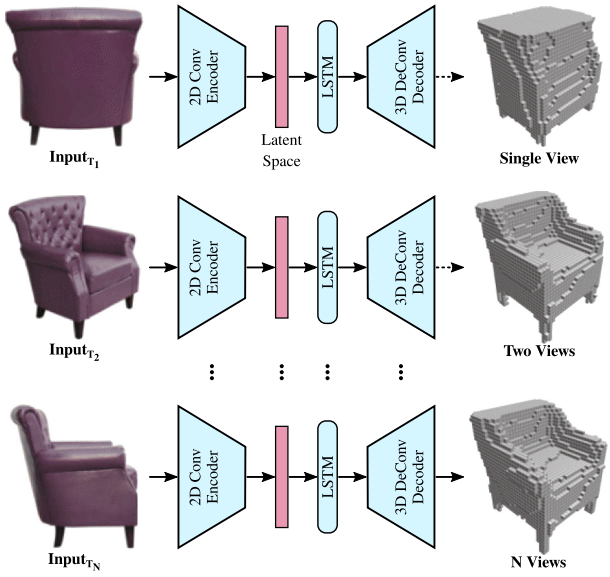


Fig. 7. *Overview of the 3D-R2N2 network [20]. The input to this network is one or more RGB images from arbitrary viewpoints, and the output is a 3-D occupancy grid. The main modules of 3D-R2N2 are an encoder, a 3-D LSTM, and a decoder.*

deconvolutional neural network (3D-DCNN) that decodes the 3D-LSTM hidden states into a higher resolution and produces the final occupancy grid.

In the LSTM module, 3D-LSTM units are located in a grid structure in such a way that each of them focuses on reconstructing a particular part of the output. Two versions of 3D-LSTMs, 3D-LSTMs without output gates and 3-D gated recurrent units (GRUs), were tried out in 3D-R2N2, in which the latter achieved better results. The output size is 32^3. Although the generation of detailed and thin parts of the objects and the reconstruction of objects with high texture levels are very challenging, 3D-R2N2 performs better than the category-specific approach proposed by Kar et al. [133], which learns 3-D shapes using camera viewpoint estimations together with object silhouettes, in SVR using real-world images. 3D-R2N2 is also able to produce accurate outputs compared to the MVS method [86] in multiview reconstruction (MVR).

Brock et al. [21] investigated generative and discriminative voxel modeling with deep ConvNet architectures. In short, their method presents a voxel-based variational autoencoder (VAE) [134], [135] for reconstruction and interpolation, a graphical user interface for investigating the latent space of autoencoders (AEs), and a deep voxel-based CNN for object classification. The output size of the network is 32^3. The voxel-based VAE learns to reconstruct features of an object, attaining acceptable reconstruction accuracy. It further facilitates the transition from one object to another by interpolating between their reconstructions. The neural model has significantly fewer parameters than FusionNet [129], i.e., 18 million as opposed to 118 million. Nevertheless, it achieves competitive results compared to ORION [127] considering that ORION uses orientation augmentations to improve the classification.

The TL-embedding network [22] learns a vector representation of an object, which is both generative in 3-D, i.e., able to reconstruct objects in 3-D space from this representation, and predictable from 2-D images, i.e., able to extract this representation from images. As shown in Fig. 8, this architecture is composed of a convolutional network, which brings about predictability, and an autoencoder, which results in generativeness. It generates outputs with 20^3 resolution. This method captures stylistic details better than the method proposed by Kar et al. [133].

Wu et al. [23] introduced a framework, called 3-D generative adversarial network (3D-GAN), which generates novel volumetric 3-D objects from a probabilistic latent space. 3D-VAE-GAN, an extension of 3D-GAN, provides the ability to reconstruct surfaces from input images. For generation and recognition of 3-D objects, this method utilizes both general-adversarial modeling [136], [137] and volumetric convolutional networks [98], [126], as illustrated in Fig. 9. Furthermore, it fuses 3D-GAN with a VAE [134] for 3-D object reconstruction from a single 2-D image. The resolution of its final output can reach up to 64^3. The classification accuracy of this network is roughly similar to volumetric learning-based approaches, such as VoxNet [126] and ORION [127], but is lower than the method proposed by Qi et al. [128]. It shows higher AP for voxel prediction compared to the work by Girdhar et al. [22] in a single-image 3-D reconstruction task. However, 3D-VAE-GAN usually creates a noisy and incomplete output from an input image. Studies conducted by Wu et al. [138] showed that, ultimately, training GANs together with recognition networks can lead to high instability.

Stutz and Geiger [25] introduced a learning-based approach with weak supervision for 3-D shape completion. It takes a 3-D bounding box and an incomplete point cloud as input and predicts the complete object shape. The completion process is done in two steps. 1) A shape prior is learned, i.e., a VAE is employed to learn a 3-D

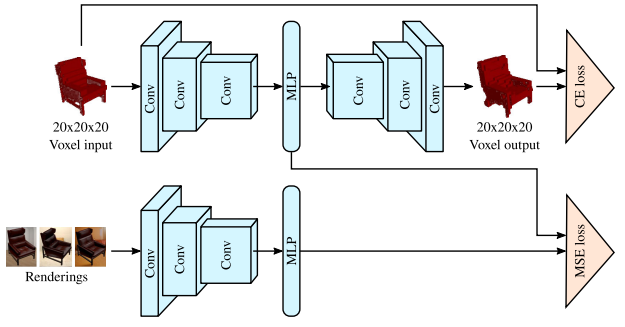


Fig. 8. *TL-embedding network [22]. During training, two types of input are fed to the network: 2-D RGB images as the input to ConvNet at the bottom and 3-D voxel maps as the input to the autoencoder at the top. The network outputs a 3-D voxel map.*

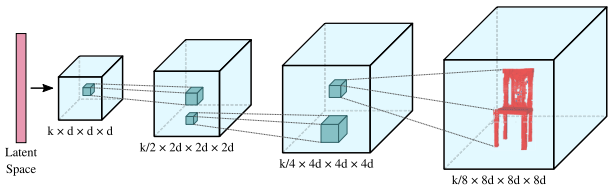


Fig. 9. *Generator architecture in 3D-GAN [23]. Five volumetric fully convolutional layers of kernel sizes 4 × 4 × 4 and strides 2 make up the generator. The discriminator architecture is usually mirroring the generator architecture.*

shape model on synthetic data, encoding shape models in a dataset using occupancy grids and SDFs at 24 × 52 × 24 resolution. 2) Shape inference is performed. For this, 3-D shape completion is considered a maximum likelihood (ML) problem. The authors used the amortized ML (AML) approach that works over the lower dimensional latent space z from the first step. It keeps the pretrained decoder from the previous step fixed and adds a new encoder. The encoder is trained without supervision, i.e., without using explicit labels, and learns to directly predict ML solutions from incomplete input observations using ML loss. The presented method was shown to be faster than a fully supervised baseline while using 9% or less supervision while being able to produce competitive results.

Dai et al. [24] fused a volumetric DNN with a 3-D shape synthesis procedure to complete partial 3-D inputs. Their approach generates the output in two major stages. 1) A shape prediction method, which predicts a volumetric grid with 32^3 resolution as a low-resolution global structure of the input. The proposed network, the 3-D-encoder-predictor network (3D-EPN), consists of 3-D convolutional layers and attempts to predict distance field values for missing data. 2) A patch-based 3-D shape synthesis method, which employs a synthesis procedure to improve local details and create a high-resolution output using CAD model priors. Given the predicted coarse output from the first stage, the authors carried out a search for similar 3-D shape models in the ShapeNet [96] database. Based on the results, they sought to find similar local patches in these shape models for the purpose of local detail synthesis. The resolution of the final voxel grid is 128^3. Without the synthesis step, 3D-EPN provides only low resolution and is unable to predict local details and fine structures. Nevertheless, it outperforms 3-D ShapeNets [98] and Poisson methods [8], [9].

In another approach, Dai et al. [139] suggested sparse generative neural networks (SG-NNs), which is a self-supervised scene completion approach that accepts an incomplete RGB-D scan as input and predicts a high-resolution 3-D reconstruction while also inferring unseen, missing geometry. The self-supervised nature of this technique allows for training entirely on real-world, partial scans. This eliminates the requirement for synthetic ground truth. Self-supervision is achieved by removing some frames from a given (incomplete) RGB-D scan, resulting in an even more incomplete input; this input is used to create an input-target pair (the original scan is considered the target scan). The difference in partialness is then correlated in this input-target pair, while regions that have never been observed are masked out during training. Despite the fact that fully complete scenes are not used as samples during training, this approach generates high levels of completeness by learning to generalize completion patterns across the training set. Dai et al. also proposed an SG-NN, a fully convolutional encoder–decoder architecture, capable of predicting high-resolution final geometry as a sparse TSDF representation. This end-to-end formulation generates a 3-D scene in a coarse-to-fine manner. SG-NN is built upon sparse convolutions [140] that operate only on surface geometry. This self-supervised approach produces more accurate and complete scenes in comparison to a fully supervised approach, such as 3D-EPN [24].

In general, voxel-based methods encounter a number of difficulties. Information loss may occur due to discretization and transformation of input data to coarse voxels. Moreover, cubic growth in memory limits the resolution and the overall computational demands bring about coarse final outputs. Generating higher resolution surfaces requires deeper networks. However, the network depth is constrained by the available GPU memory. Therefore, it may affect the ability of CNNs with volumetric decoders in producing high-resolution outputs [141].

2) Octrees: Dense voxel representations are associated with a number of challenges regarding resolution, memory, and computational complexity. In many cases though, the 3-D shape surface occupies only a small portion of 3-D space. Hence, octrees mark a popular approach for partitioning space, as they allow for the 3-D data to be stored in a sparse structure [142], [143]. For octree construction of a 3-D shape, a bounding cube is created around the entire shape. This bounding cube will be recursively subdivided. In each step, all cuboids, which are occupied by a shape boundary, are traversed, and each of them is divided into eight smaller, equally sized cuboids. However, in order to enable CNN operations on an octree, this data structure needs to be updated and slightly changed, which leads to complex implementations, while the resolution is still limited by the underlying 3-D grid [45]. Hence, convolutions and pooling to octrees are applied similar to CNN operations on dense voxels with the main difference being that the elementary operand is an octant.

a) OctNet: Riegler et al. [144] presented OctNet that enables the usage of high-resolution inputs for DL purposes. OctNet is based on a 3-D-CNN that can be applied to a special form of octree data structure to learn representations from high-resolution 3-D data. Vanilla octree implementations might encounter data access speed issues in high-resolution (high recursion depths) octrees. On the other hand, for convolutional network operations, such

as convolution or pooling, it is crucial to have frequent access to different data elements, such as cell neighbors. In order to provide faster data access and reduce cell traversal time, the authors proposed a hybrid grid-octree data structure. They used a shallow octree, which is an octree with maximum depth $D = 3$, as a basic building block. Several of these shallow octrees are stacked in a regular grid structure to cover the whole volume. Input resolution effects of this representation were evaluated on three different tasks: 3-D classification, 3-D orientation estimation of unknown object instances, and semantic segmentation of 3-D point clouds. For high-resolution inputs in the 3-D shape classification task, OctNet runs faster and requires less memory as opposed to DenseNet, a densely voxelized version of OctNet. In general, both OctNet and DenseNet perform better than a shallow network such as VoxNet [126], verifying that network depth is of great importance.

OctNet does not generate an octree structure, and this structure has to be known in advance for both input and output. In classification and semantic segmentation tasks, this does not comprise a problem. However, learning the volumetric structure of objects and scenes, and being able to construct them is crucial in generative tasks, such as reconstruction, generation, and completion, since the input and output partitioning structure might be different. OctNetFusion [26] proposes a learning-based approach, which learns to partition the space and can predict an SDF or a binary occupancy map. The network takes one or more 2.5-D depth maps as input. To reconstruct precise and complete 3-D outputs, it fuses depth information from different viewpoints into a coarse volumetric grid. Then, this volumetric grid (grid-octree structure) is fed to the OctNetFusion network architecture, consisting of encoder–decoder modules. The network determines whether a cell should be subdivided or not in a coarse-to-fine manner. The output resolution can be up to 256^3. This approach performs qualitatively and quantitatively better than traditional volumetric fusion approaches, such as vanilla TSDF fusion [124] and TV-L1 fusion [145] for volumetric fusion tasks and Voxlets [146] for volumetric shape completion from a single image.

b) O-CNN: Another concurrent work in the scope of octree-based CNNs (O-CNNs) for 3-D shape analysis is the O-CNN [147]. The authors' main idea is to represent 3-D objects with octrees and execute 3-D-CNNs only on nodes or cuboids, which are occupied by boundaries of the 3-D object, instead of sliding the convolutional kernel over the whole voxel grid, as done for the standard convolution computation in full voxel grids. The network constructs an octree from an input-oriented 3-D model, e.g., an oriented triangle mesh or a point cloud with oriented normals, and enriches each octant of this data structure with metainformation, such as shuffle key vectors, label vectors, and input signal, which are needed for the convolution operations. Furthermore, a hash table is built to accelerate neighborhood search in the convolution. By storing the octree data structure in the graphical memory, O-CNN can be easily and efficiently trained and evaluated on GPUs. To demonstrate the efficiency of their network, the authors evaluated it on three shape analysis tasks: object classification, shape retrieval, and shape segmentation. In terms of classification accuracy, O-CNN performed better than VoxNet [126], slightly worse than the method proposed by Brock et al. [21], and competitive to nonvoxel-based methods, such as PointNet [148]. In addition, the impact of different input representations on the same network architecture (O-CNN) was investigated. Results showed that an octree input achieves higher accuracy compared to full voxel structures. For object part segmentation, O-CNN yields better or comparable performance than other methods, such as PointNet [148].

To improve the computation and memory efficiency of O-CNN, Wang et al. [27] proposed the extension "Adaptive O-CNN," which consists of an encoder–decoder structure and uses patch-guided adaptive octree shape representations. Contrary to approaches such as volumetric-based CNNs, where the output is generated as voxels of the same resolution, this method can generate adaptive octrees based on a patch-guided partitioning strategy and with differently sized planar patches. The underlying assumption is the subdivision rule, which states that splitting all octants to the finest level is not necessary. The process can be stopped early for some of the octants, and the local shape inside these octants can be represented by simple patches, e.g., planar patches. However, this approach limits the quality of the output and may encounter some difficulties in generating watertight and curved surfaces. Adaptive O-CNN obtains better or comparable classification accuracy than PointNet [148], OctNet [144], and O-CNN [147], yet it performs worse than PointNet++ [149], Kd-Network [150], and the method proposed by Brock et al. [21]. For the task of shape reconstruction from a single image, Adaptive O-CNN surpasses PointSetGen (PSG) [30] and AtlasNet [34] in generating more detailed geometry.

c) Other octree prediction approaches: Häne et al. [28] introduced a hierarchical surface prediction (HSP) framework for high-resolution voxel grid prediction in 3-D object reconstruction. The main idea boils down to generating and predicting high-resolution voxels around the predicted surface and coarse-resolution voxels for the interior and exterior parts of an object. The high-resolution voxels are not predicted directly, but, instead, a coarse-to-fine approach is used to create smoother 3-D models hierarchically and in a multiresolution fashion. Starting with approximating the coarse geometry of the output, more finely resolved details are added step by step by refining the surface. This process, finally, results in a voxel grid with up to 256^3 resolution. The proposed method is based on an encoder–decoder architecture. A convolutional encoder encodes input to a feature vector, and then, an upconvolutional decoder predicts the voxel grid or final data structure (called voxel block octree

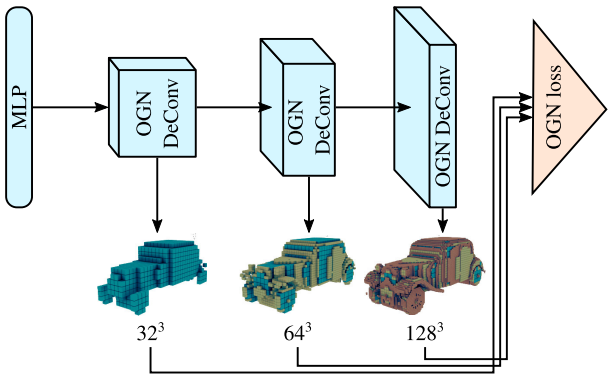


Fig. 10. *OGN [29] takes an input 3-D shape and gradually reconstructs octrees as the output in different resolutions.*

data structure in this article). Classifying each voxel as the boundary, free space, or occupied space, only voxels with a boundary label require high-resolution prediction since they cover the actual surface. The major difference between HSP and OctNet [144] is that OctNet takes the structure of the shallow octrees as input, while HSP predicts the structure of the tree together with its content. HSP produces more accurate surfaces with higher resolutions compared to low-resolution baselines predicting dense voxels.

In a similar approach, Tatarchenko et al. [29] suggested an octree generating network (OGN) that is a convolutional decoder that can generate and predict the octree structure of 3-D shapes, along with the occupancy value of each cell. It operates on octrees and reconstructs 3-D shapes in a multiresolution manner, as illustrated in Fig. 10. This method generates results up to a resolution of 512^3. The network gradually reconstructs a high-resolution surface from the initial, low-resolution dense voxel grid using hash-table-based octree blocks. If the reconstructed surface has not yet reached the final output resolution, cells with a "mixed" state, i.e., undetermined state, will be passed to the next layer of the network for further subdivision. Providing the same accuracy as dense voxel grids in low resolutions, OGN offers less memory consumption and shorter run-time in higher resolutions in comparison to voxel grid-based networks. In particular, it is 20 times faster and requires two orders of magnitude lower memory usage at 512^3 resolution.

B. Point-Based Representations

These days, point clouds are becoming increasingly important and available due to the improvements in scanning devices in recent years. A point cloud is a set of points in 3-D space, inferred by various 3-D data acquisition techniques. It is an irregular data format since there is no canonical order between the points in a set. Each point can be defined by its (x, y, z) coordinates. Therefore, the size of the matrix representing a 3-D object is initially $N \times 3$ for N points. The number of columns in this matrix representing the features might be extended if other information, such as color and normal, exists. Considering the irregular and unordered nature of point clouds, it is difficult to apply DL techniques, such as CNNs, directly on them. Consequently, in order to process a point cloud with neural networks, it was common to transform them into voxel grids or collections of images. These transformations usually present numerous challenges, such as information loss, voluminous data, resolution constraints, and high computational costs. To reduce the overhead of data transformation to other data formats, different methods for effectively processing point clouds with neural networks have been proposed, which will be discussed in Sections VI-B1 and VI-B2.

1) PointNet and PointNet++: Pioneer works in the field of learning global features directly on point clouds are PointNet [148] and PointNet++ [149]. PoinNet as proposed by Qi et al. [148] directly consumes a raw point cloud as an input and uses it for discriminative DL tasks, e.g., object classification, semantic segmentation, and part segmentation. As illustrated in Fig. 11, each of the points in the input set is processed by a small neural network individually and independently based on its own coordinates, resulting in a high-dimensional embedding of the points. Following the embedding step, a simple symmetric function, such as max pooling, is utilized to aggregate the encodings from each of the points. The symmetric function is chosen such that it pays attention to the permutation invariance of the input points. The aggregation step brings about a global feature vector, which encodes the whole shape and can be fed to different neural networks for recognition purposes. PointNet achieves higher classification accuracy compared to the LFD approach [122], which is a 3-D model retrieval method, SPH [130], and other methods with volumetric representation, such as 3-D ShapeNets [98], VoxNet [126], and another method previously proposed by Qi et al. [128]. Although it has around 17 times fewer parameters than multiview-based methods, such as MVCNN [131], its performance is only slightly lower compared to these methods. PointNet pro-

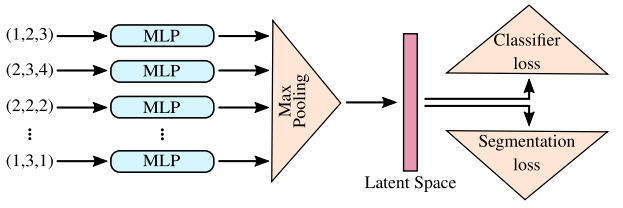


Fig. 11. *PointNet architecture [148], which is used for classification and segmentation tasks, directly accepts a point cloud as input. Each of the points in the input point cloud is processed by a small neural network individually and independently. Then, point features are aggregated by max pooling, a simple symmetric function that respects the permutation invariance of the input points. The aggregation step creates a global feature vector that encodes the entire shape.*

vides linear complexity $O(N)$ in both spatial and temporal domains, where N is the number of input points, while the complexity grows squarely with respect to image resolution for multiview methods and cubically with respect to the volume size for volumetric methods. More importantly, due to it satisfying the permutation invariance condition, PointNet cannot capture local information and, thus, lacks generalization.

In order to resolve the issues of PointNet, Qi et al. [149] introduced the extension PointNet++, which pays more attention to local features and combines them with global features to infer better results. The architecture is built on top of PointNet, enriching it with a hierarchical feature learning approach. The whole process, which is done recursively, can be summarized as follows: 1) specifying centroids of local regions by sampling a subset of the input point cloud using the farthest point sampling (FPS) algorithm; 2) finding local neighborhoods of these centroid points using radius-based ball query; 3) applying a mini-PointNet in each neighborhood to mimic the concept of a convolution kernel and conduct convolution-like operations in point space for the purpose of local feature extraction. The presented method proved to be robust toward nonuniform sampling density, which might occur due to perspective effects, variations in radial density, motion, and so on. Compared to PointNet, PointNet++ has an improved classification accuracy for the ModelNet40 dataset.

2) Point cloud reconstruction and generation: PointNet was mainly implemented for discriminative tasks, such as classification and segmentation. The first approach for reconstructing a 3-D point cloud of an object from a single (monocular) RGB or RGBD image was proposed by Fan et al. [30] and is based on a generative learning-based approach. The main contributions of this work are given as follows: 1) designing a point set generator network; 2) proposing two proper loss functions for the comparison of the ground truth with the network's predictions for point sets, i.e., CD and EMD; 3) modeling uncertainty and ambiguity of the ground truth. The proposed network is composed of an encoder and a predictor part. The encoder transforms the input into an embedding space. The predictor is divided into two parallel branches: a deconvolution (deconv) branch and a fully connected (fc) branch. The deconvolution branch learns the smooth parts and main body of the object, while the fc branch learns nonsmooth parts and details. The results of these branches are then concatenated to create the final point set. In comparison to 3D-R2N2 [20], which generates a volumetric representation from single or multiview images, this method produces better results on CD, EMD, and IoU metrics. In addition, it is able to reconstruct thin structures more accurately.

Achlioptas et al. [31] proposed a solution for generative tasks and unsupervised representation learning based on an end-to-end pipeline that can reconstruct point clouds using deep autoencoders (AEs) and GANs. The autoencoder extracts features by learning a lower dimensional representation of the input, based on which the GAN [136] generates point clouds. In the autoencoder architecture, the authors exploited a PointNet-like encoding scheme to learn compact representations. The encoder generates a latent code that is invariant to the order of input points. The latent code is converted back to a point cloud using a standard deep network with three fc layers as a decoder. The authors further investigated three different approaches for point cloud generation: 1) GAN operating on raw point cloud; 2) latent-GAN, which is a plain GAN being trained on the latent space of the pretrained AE; 3) Gaussian mixture models operating on the latent space learned by AE. The study indicated that the proposed AE provides good generalization capacity toward unseen data. However, the output of the proposed DL model architecture is limited to 2048 points, and generating high-quality surfaces with such a small number of points is challenging.

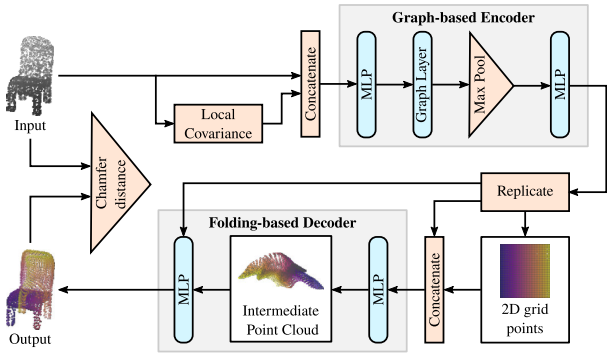


Fig. 12. ***FoldingNet architecture [32] consists of a graph-based encoder (an improved and generalized version of PointNet), which encodes local neighborhood structure information, and a folding-based decoder, which reconstructs the point cloud from a 2-D grid template deformation process.***

Another closely related approach that attempts to solve unsupervised learning challenges using deep autoencoders is FoldingNet [32]. The presented architecture, as illustrated in Fig. 12, utilizes a simple graph-based scheme as the encoder part (similar to the method proposed in [151], an improved and generalized version of PointNet) in order to encode local neighborhood structure information. Since applying convolution operations on graphs is difficult, the authors suggested building the k-nearest neighborhood graph (K-NNG) and repeatedly applying max-pooling operations on each node's neighborhood. This way, the DL model is able to capture locality and extract features of neighboring points. For the decoder part, a folding-based scheme is proposed to reconstruct the point cloud from a 2-D grid template deformation process. Due to the fact that 3-D point clouds are often sampled from object surfaces, one can make the assumption that any 3-D object surface can be converted and squeezed into a 2-D plane. It is also possible to reverse this process, i.e., wrapping 3-D shapes with a fixed 2-D paper (plane). This property builds the foundation of the proposed method.

The decoder maps 2-D points from a 2-D template grid to the surface of the 3-D object using folding operations. The definition of the folding operations, i.e., 2-D-to-3-D mapping, is the main contribution of this article, making it the first single learned parametric function embedding from a (gridded) 2-D (point) manifold into 3-D space and a fundamental building block for other surface reconstruction approaches. FoldingNet's decoder requires about 7% of the parameters of the fc decoder proposed by Achlioptas et al. [31], which is significantly smaller than the latter. However, it was shown to perform better at feature extraction in terms of classification accuracy and reconstruction loss. Overall, FoldingNet achieves higher classification accuracy than other unsupervised methods, such as LFD approach [122], SPH [130], TL-embedding network [22], and 3D-GAN [23].

PointFlow [33] is a 3-D point cloud generation framework that learns a distribution of distributions, i.e., the distribution of shapes and its respective points. A VAE is applied to transform sampled 3-D points from the point prior into a realistic point cloud conditioned on a shape vector. The distributions are modeled in two steps. First, the distribution of the latent space of shapes is learned. To enable the method to sample multiple shapes, PointFlow extracts latent vectors of different shapes. A sampled Gaussian vector (a shape prior) is transformed into a shape latent vector using a continuous normalizing flow (CNF) [152], [153], [154]. In the second step, the distribution of points on a specific shape is learned for shape generation. Given a sampled 3-D Gaussian point cloud (point prior) and a shape latent vector inferred from the first step, a CNF is used to move input points to their new location and transform them into the target shape. For generative tasks, PointFlow outperforms the methods proposed in [31] in terms of the 1-nearest neighbor accuracy (1-NNA) metric while having fewer parameters. With respect to the EMD score, it achieves better autoencoding performance compared to Achlioptas' method [31] for point cloud reconstruction from inputs.

Several recent studies have investigated point cloud upsampling [155], [156], [157], normal and curvature estimation from point clouds [158], [159], classification [160], [161], [162], [163], [164], [165], [166], segmentation tasks [160], [161], [162], [164], [166], [167], [168], object detection [163], [169], and point cloud denoising [170]. Although point-based representation approaches discretize the surface of the shape into a set of 3-D points, they do not model the corresponding connectivity. Thus, additional postprocessing steps are needed to generate the final high-quality 3-D mesh. On the other hand, existing approaches are very limited in terms of the number of generated points leading to limited output quality.

C. Mesh-Based Representations

Meshes are irregular types of data that are difficult to predict by neural networks. Their components are vertices, edges, i.e., pairs of vertices, and (triangular) faces, i.e., triplets of vertices. Therefore, researchers have investigated different paths to address mesh-based representations, namely, patch-based approaches [34], [37], deformable template-based approaches [38], [39], [40], [41], [70], [171], and other mesh generation methods [35], [36], [42], [43], [44], [172].

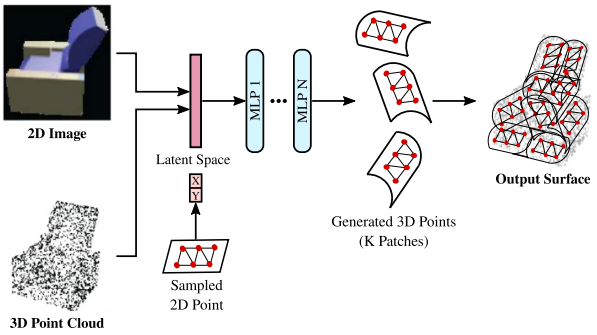


Fig. 13. *AtlasNet [34] is a patch-based approach that takes either a 2-D image or a 3-D point cloud as input and outputs a 3-D mesh. MLPs are used to estimate the target 3-D surface, which learns the local mapping of 2-D-points to 3-D-surface points.*

1) Patch-Based Approaches: Groueix et al. [34] introduced a method for 3-D surface generation, called AtlasNet, as illustrated in Fig. 13. They suggested generating a 3-D surface and representing it as a set of folded 2-D squares. The input shape can be either a 2-D image or a 3-D point cloud. The method outputs the corresponding 3-D mesh and its atlas parameterization. The main steps of the approach include encoding an input 3-D point cloud into a 3-D shape and reconstructing the 3-D shape from an input RGB image. 3-D point clouds are encoded using a PointNet-based encoder, which transforms the input point cloud into a 1024-D latent vector. Input images are encoded using ResNet-18 [173]. The decoder consists of four fc layers, which extract the final surface. The target 3-D surface is estimated using multilayer perceptrons (MLPs), which learns the local mapping of 2-D-points to 3-D-surface points. Therefore, by transforming the 2-D squares to the 3-D surface using learnable parametrizations, i.e., MLPs or patches, the final surface is covered in a way similar to putting paper strips on a shape to make a papier-mâché. The difference between the proposed method and FoldingNet [32], which is a folding-based method, is that FoldingNet deforms just one 2-D square or patch, while AtlasNet investigates a varying number of 2-D squares. Results from AtlasNet showed that the usage of multiple patches improves 3-D reconstruction. For SVR from a 2-D RGB image, AtlasNet yields qualitatively better performance compared to the dense voxel-based method 3D-R2N2 [20], the octree-based method HSP [28], and a point-based method [30]. Furthermore, it was shown that AtlasNet provides good generalization properties;

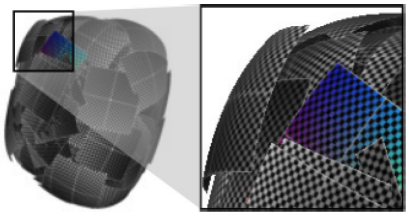


Fig. 14. *Meshlet inconsistencies adapted from the patch-based approach paper [37].*

however, it generates artifacts such as self-intersecting parts and overlapping patches.

Badki et al. [37] proposed an approach to extract a 3-D mesh from a noisy, sparse, unordered, and nonoriented set of points. Instead of learning shape priors at the object level, the method learns them locally while enforcing global consistency. In order to represent these priors and local features, small mesh patches, called meshlets, were used. These meshlets can be interpreted as a dictionary of local features and learned priors. The final mesh is the union of all meshlets. The authors used a VAE for learning the priors by using a very large dataset of meshlets, which was extracted from objects in the ShapeNet dataset. During training, the local priors are learned with meshlets. At inference, meshlets are deformed to match the input point cloud via distance minimization. Since individual meshlets are updated independently in order to adapt to the points, the overall mesh extracted from their union is not watertight. Therefore, a global consistency step is performed to eliminate inconsistencies across all meshlets, as illustrated in Fig. 14. Compared to occupancy networks [45] and AtlasNet [34], which are class-specific algorithms that learn priors at the object level, and deep geometric priors [174], this method produces better quantitative results in terms of CD and HD metrics. It also performs qualitatively well at reconstructing objects from unseen classes during training, coping with noise, and being robust to dramatic changes in the object's pose.

For all the aforementioned methods, mesh patches and the tessellation process may affect the quality of the final surface, especially for complex shapes. Therefore, these approaches may generate self-intersecting meshes and might be unable to generate closed surfaces.

2) Deformable Template-Based Approaches: Deformable template-based approaches take a template mesh with predefined interconnections as input, deform the vertices, and predict the final shape based on this. These approaches can generally reconstruct meshes and shapes with simple topology; however, they struggle to generate complex structures with a lot of details. Wang et al. [38] designed Pixel2Mesh, an end-to-end reconstruction pipeline for extracting a 3-D triangular mesh from a single RGB image. Taking an input image and an ellipsoid with fixed numbers of edges and vertices as the initial mesh, it gradually deforms the mesh using a graph-based CNN [graph convolutional network (GCN)] to generate the final 3-D shape. As illustrated in Fig. 15, the overall method is composed of two main parts. 1) An image feature network (2D-CNN), which is used to infer perceptual features using an input color image. 2) A three-block cascaded mesh deformation network (graph-based ResNet) that takes care of initial mesh deformation in a coarse-to-fine manner. Each graph-based ResNet block takes the perceptual feature concatenated with 3-D feature encoding of the input mesh as input. In their study, the authors showed that Pixel2Mesh outperforms 3D-R2N2 [20] and the point-based method proposed by Fan et al. [30] in terms of the mean of F-score, CD, and EMD metrics. Qualitywise, it produces smoother surfaces with local details. Nevertheless, the approach shows generalization issues and can only generate meshes and objects of topologies similar to the initial mesh.

Pixel2Mesh++ [39] works along with Pixel2Mesh to produce 3-D meshes from multiview images. The main idea is that adding more images (three to five) of an object as input provides more information for a shape generation method and, thus, results in more accurate and detailed reconstructions. Pixel2Mesh++ consists of a multiview deformation network (MDN), which processes cross-view information for the prediction of optimal deformations. First, a coarse mesh is produced by Pixel2Mesh, which is then fed to the MDN part to be refined progressively by adding details. With regard to the F-score metric, Pixel2Mesh++ generates better results than 3D-R2N2 [20], learned stereo machine (LSM) [175], and two other baselines that the authors implemented using Pixel2mesh [38]. In addition, it generalizes well across various semantic categories and produces high-quality outputs with accurate details.

Recent efforts by Kanazawa et al. [40] utilized a CNN image encoder followed by three modules for 3-D shape generation, camera pose estimation, and texture prediction. The CNN acts as an encoder, producing a latent representation of a single input image, which is fed to the three prediction modules. The 3-D structure of a shape is generated by deforming a learned category-specific mean shape with instance-specific predicted deformations. Texture is parameterized as a UV image that is predicted using texture flow. This mechanism enables the method to transfer the texture of one instance onto another. However, it cannot produce the detailed structure of the input shape. The presented approach obtains comparable results to the one proposed by Kar et al. [133] in terms of the IoU metric. Kar et al. [133] exploited segmentation masks and optionally a set of keypoints as annotations during inference to generate 3-D rigid objects. Contrary to that, the method of Kanazawa et al. [40] only utilizes these annotations during training and directly predicts a 3-D structure form an unannotated input image at inference time.

Hanocka et al. [41] introduced Point2Mesh for reconstructing meshes from point clouds. The core idea is a mesh fitting process for the reconstruction of the final mesh.

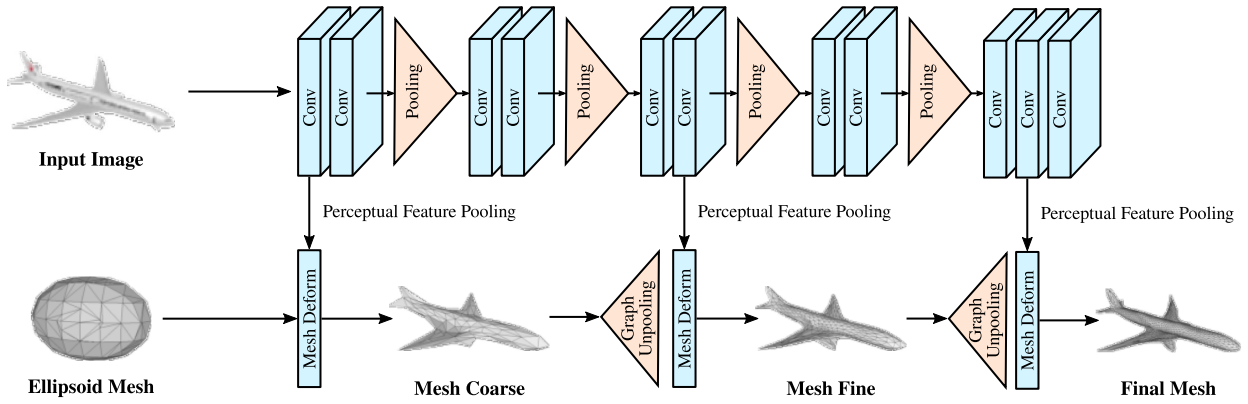


Fig. 15. *Pixel2Mesh network [38] is a deformable template-based approach that reconstructs a 3-D triangular mesh from a single RGB input image. It consists of three mesh deformation blocks used for mesh resolution enhancement and vertex location estimation.*

In addition to the input point cloud, an initial watertight mesh is fed to the network. This initial mesh represents a coarse approximation of the point cloud, which is iteratively deformed from outside-in using a CNN to fit the input point cloud, as illustrated in Fig. 16. Accordingly, a network learns displacement and deformation of the mesh vertex positions. The optimization of Point2Mesh is based on MeshCNN [176], which is a CNN-based pipeline applied on triangular meshes. Unlike Screened PSR, Point2Mesh is agnostic to normal orientation and ensures watertight reconstructions from noisy input with missing parts and unoriented normals. It also achieves a higher F-score compared to Screened PSR [9] and deep geometric priors [174] for shape denoising and completion. However, Point2Mesh requires a large amount of compute time and memory, possibly alleviated by data parallelism or model parallelism [177].

3) Other Mesh Generation Methods: Liao et al. [42] investigated end-to-end 3-D surface prediction using a differentiable MC (DMC) algorithm. In previous research, the surface prediction was solved in two steps: first, predicting an intermediate SDF/occupancy representation using an auxiliary loss, and second, taking a postprocessing step for 3-D mesh extraction separately, such as the MC algorithm. On the other hand, applying backpropagation to the MC algorithm is intractable due to nondifferentiability. Hence, in order to unite these steps to create an end-to-end framework, the authors inserted a differentiable formulation as a final layer into a 3-D-CNN. A point cloud, which is used as input, is directly converted into a volumetric representation using a grid pooling operation, e.g., max pooling in each cell. An encoder–decoder network with skip connections is then used to process pooled features, with the decoder operating in volumetric space. That way, it not only estimates occupancy probabilities but also predicts the vertex displacement field for a surface mesh. Compared with baseline methods that infer occupancy or TSDF first and then apply MC as a postprocessing step, DMC achieves superior results with respect to CD, accuracy, and completeness metrics. Nevertheless, difficulties may arise while reconstructing very thin surfaces, and disconnected parts can become connected.

Fig. 16. *Point2Mesh [41] takes a point cloud (in blue) and a deformable initial mesh as input and gradually reconstructs the final output shape.*

Scan2Mesh [43] is a generative model that combines convolutional and graph neural network architectures to predict a complete, lightweight, and structured 3-D mesh representation from an unstructured and incomplete range scan of an object. The aim is to predict both vertex location and edge. Initially, the feature space is computed through a set of 3-D convolutions from input TSDF. The vertices are then predicted based on the extracted features. An fc graph is generated from the predicted vertices, and all of the vertices are connected to each other via edges. Next, a graph neural network is used to classify edges and extract the ones that belong to the mesh graph structure. Using this intermediate graph of predicted edges and vertices, a dual graph is created which comprises a set of valid potential faces. Finally, another GNN is applied to predict the final face structures from the dual graph. Scan2Mesh offers better qualitative and quantitative performance compared to 3-D ShapeNets [98], 3D-EPN [24], and PSR [8], [9]. However, it depends on fc graphs for predicting edges, which leads to limitations in model size (MS).

Mesh R-CNN [44] is an approach that unifies both 2-D perception and 3-D shape prediction. It takes a single RGB image as an input, detects 2-D object instances in the image, and creates a category label, bounding box, segmentation mask, and 3-D mesh predictions of the detected objects as the outputs. Mesh R-CNN utilizes

Mask R-CNN [178], an end-to-end region-based 2-D object detector, for the detection of 2-D objects. The 3-D shape prediction step depicted in Fig. 17 is based on a hybrid approach, which primarily produces a coarse voxel representation of a detected object, transforms this voxelization into an initial 3-D triangular mesh, and, finally, refines this mesh by modifying the vertex positions using a GCN. This approach achieves better results compared to a voxel-based method, such as 3D-R2N2 [20], a point-based method [30], and a mesh-based method, such as Pixel2mesh [38] in single-image shape prediction considering CD and F1-score metrics.

Liu et al. [35] attempted mesh reconstruction from input point clouds by fully utilizing the input and simply adding connectivity to the existing points. Toward this end, they introduced a deep point cloud network that proposes candidate triangles and predicts faces. This information is provided as input to a mesh generation module. First, a k-nearest neighbor (k-NN) graph is built for each point in the input point cloud, in order to decide which three points should form a triangle face and infer candidate triangle proposals. Next, an MLP network is employed to classify candidate triangles and filter out incorrect triangles, such as the ones that connect two independent but spatially adjacent parts of the shape, using the intrinsic–extrinsic ratio (IER). To infer the local connectivity between vertices comprising a triangle, the ratio of the geodesic distance (intrinsic metric) and the Euclidean distance (extrinsic metric) was proposed. Finally, in a postprocessing step, the remaining candidate triangles are sorted and merged in a greedy way to generate the final mesh. The approach outperforms several learning-based methods, such as AtlasNet [34], deep geometric priors [174], deep MC [42], and DeepSDF [50], as well as traditional reconstruction methods, such as PSR [8], [9], MC [125], and BPA [10] in terms of F-score, CD, and NC metrics. Moreover, it generates higher quality outputs with fine-grained structures than the aforementioned methods and offers the capability to be transferred to unseen categories.

Daroya et al. [36] proposed a recurrent neural network (RNN)-based method, called recurrent edge inference network (REIN), to produce triangulated surface meshes from sparse input point clouds using a bottom-up approach. The network tries to predict edges sequentially and generates a mesh by processing points one at a time from a queue of points. The latent vector of the input point cloud, which is inferred by a PointNet-based [148] autoencoder, is also used to enrich the data with global structure information of an object. For edge prediction, the authors relied on the application of recurrent networks, inspired by GraphRNN [179]. An RNN can be a good choice for inferring sequential predictions based on previous states [180]. To tackle memory issues of processing large point clouds, small sections of the input point cloud are fed into the network one at a time, instead of processing all of it at once. In each small section, points in the queue are processed consecutively by REIN in two steps. 1) *Edge Prediction:* REIN tries to predict connections, i.e., edges, between the new vertex (which was chosen from the queue) and the current partially predicted mesh. Two RNNs are used for edge prediction: State RNN and Edge RNN. State RNN encodes the current state of the graph with its nodes and edges, given a point cloud and its latent vector as input. Edge RNN attempts to predict the sequence of edges considering the current state. 2) *Face Generation:* All of the vertices and predicted edges are investigated to form faces. However, the face generation module encounters problems generating surfaces from edge predictions, especially for nonmanifold surfaces. Qualitatively and quantitatively, REIN produces better mesh surfaces than BPA [10] and PSR [8].

D. Implicit Neural Representation

Neural networks are universal function approximators [181]; hence, they can be used to approximate any measurable function, including SDF and occupancy/indicator function, or to model other properties, such as radiance fields. Neural networks that parameterize such implicitly defined functions, without explicitly parameterizing the surface or properties of interest, are considered implicit neural representations [51].

Similar to implicit functions stored in discretized voxel grids, different functions can provide geometric information for parameterizing a surface by a neural network [123]. There are also other functions that focus on capturing surface-related properties, such as appearance, texture, or reflectance properties. In particular, these functions can be as follows.

1) Level set methods define a DF f on the entire point set and then extract the zero-level set $f = 0$ as the boundary of an input object, as illustrated in Fig. 18. They divide a 3-D space threefold into an interior part, an exterior part, and an exact overlap with the object's surface. Given a point (x, y, z), the function f calculates the distance of this point to the boundary of the object, specifies its sign (SDF) [182], and decides the location of the point w.r.t. the surface. The sign indicates whether a point is inside or outside of the surface. Therefore, in contrast to SDFs stored in voxels that discretize 3-D space and store SDF value in each voxel, SDFs in implicit neural representation are calculated for each point individually using a neural network. DeepSDF [50], which will be explained further down in this survey, was the first paper to propose this approach.
2) Occupancy functions model an approximate likelihood of whether a point is occupied by part of an object or not. This can be expressed as a binary classification problem to classify a point as occupied or unoccupied. The approach can be interpreted as a special case of SDF that only considers the sign of SDF values [50]. Occupancy networks [45] and

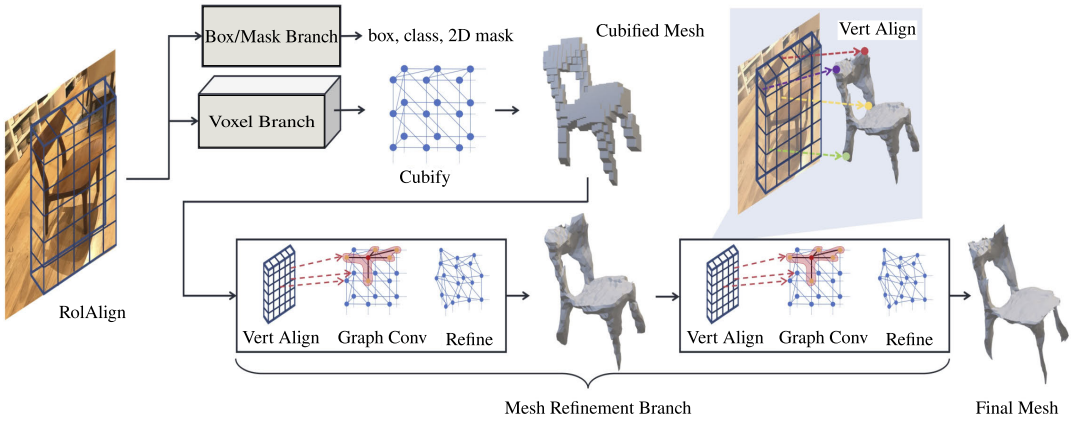


Fig. 17. *Mesh R-CNN [44] architecture. After the object detection step, the voxel branch predicts a coarse voxel representation for each object detected by Mask R-CNN [178]. Then, in the mesh refinement branch, the cubified object is transformed into the mesh after a series of refinement steps.*

IM-NET [49] fall into this category and will be clarified subsequently.

3) Radiance fields refer to a set of techniques that aim to model the radiance or appearance properties of an object or scene. Notable examples of these methods include NeRF [57] and its variants, and Sections VI-D2a and VI-D2b will provide thorough explanations of them.

1) Implicit Neural Representation Based on Variants of SDF or Occupancy Function: The key idea behind these implicit neural representations is to represent a shape as a neural network that takes a point in space as input and outputs some property of that space, i.e., mapping it to occupancy or signed distance of the shape at that coordinate. However, implicit neural representations cannot directly derive detailed 3-D shape features. Thus, an extraction step is needed to infer a corresponding explicit representation, such as a mesh. A possible isosurface extraction approach is the classical MC algorithm [125].

Compared to voxel-based representations, the memory cost of implicit neural representations remains constant with respect to the resolution. However, the capability of these methods to reconstruct fine details is constrained by the capacity of their underlying network architectures [51].

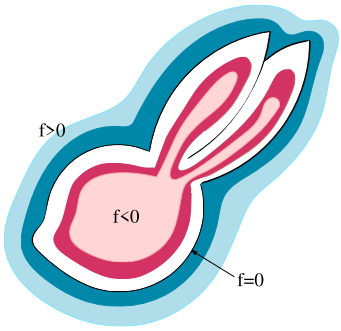


Fig. 18. *Level set methods divide a 3-D space into three parts: an interior part ($f < 0$), an exterior part ($f > 0$), and an exact overlap with the object's surface ($f = 0$).*

As mentioned previously, occupancy networks, IM-NET, and DeepSDF [45], [49], [50] represent pioneer works in implicit neural representation concurrently. Mescheder et al. [45] introduced a new representation for 3-D geometry, called occupancy networks, which can predict the continuous occupancy function using a neural network for the extraction of 3-D meshes. As illustrated in Fig. 19, the occupancy function is approximated with a DNN that determines an occupancy probability value between 0 and 1 for every possible point in 3-D point space (similar to a neural network for binary classification). The mesh is then generated from the occupancy network by utilizing a simple multiresolution isosurface extraction (MISE) algorithm, which employs octree structures and the MC algorithm [125]. This expressive approach does not require the discretization of 3-D space. The representation can be inferred from different kinds of input, such as single images, noisy point clouds, and coarse discrete voxel grids, and can encode various structures efficiently. In comparison to methods using different 3-D representations, such as 3D-R2N2 [20] (a voxel-based method), point set generating networks [30] (a point-based method), and Pixel2Mesh [38] and AtlasNet [34] as mesh-based techniques, occupancy networks show competitive qualitative and quantitative results for various inputs, e.g., single images, noisy point clouds, and coarse discrete voxel grids.

In a similar fashion, Chen and Zhang [49] attempted to solve 3-D shape analysis and synthesis problems by proposing an implicit field decoder (IM-NET), which is based on the application of binary classifiers. Based on two inputs, a point coordinate and a feature vector encoding a shape (extracted from a shape encoder), IM-NET specifies whether the point is inside or outside the surface, using only the sign of its SDF. They utilized their proposed implicit decoder as the decoder part of some conventional frameworks (such as autoencoders (AEs) and GANs) and

proposed IM-AE and IM-GAN, respectively. IM-AE and IM-GAN can be used for both 3-D reconstruction and shape generation tasks. Based on visual results, IM-AE generates smoother and high-quality surfaces compared to a classical 3-D-CNN-based decoder implementation, operating on voxelized shapes. IM-GAN showed better performance compared to AtlasNet [34] (in which output quality is constrained by the number of generated points) and 3D-GAN [23] (low coverage). For the single-view 3-D reconstruction task, the proposed framework constructs higher quality results than AtlasNet [34] and HSP [28]. However, applying the implicit decoder on each point in the training set increases training time considerably. In addition, the network does not generalize well to other categories since it is trained individually for each shape category.

With DeepSDF [50], a novel shape representation based on the concept of SDFs was introduced. Instead of storing SDF in a discretized regular grid, as done in classical surface reconstruction techniques, the network directly learns continuous 3-D models of SDF from point samples. The trained network predicts the corresponding SDF value of the input data, from which the zero-level set surface can be extracted. The zero isosurfaces can be rendered and visualized through raycasting or polygonization algorithms, e.g., MC [125]. The network takes (x, y, z) coordinates and a shape encoding vector as input to model a dataset of shapes. In order to obtain a meaningful latent space of shapes, an autodecoder is used for learning a shape embedding without an encoder. One of the advantages of the method is that the network size is considerably smaller compared to the voxel-based methods. DeepSDF outperforms Atlasnet [34] (a mesh-based method) and OGN [29] (an octree-based method) in reconstructing complex topologies with fine details. It further outperforms 3D-EPN [24] (SDFs stored in voxels) for the shape completion task.

Sitzmann et al. [51] introduced a novel architecture, called sinusoidal representation networks (SIRENs), an fc neural network that uses periodic sine as its nonlinearity for implicit neural representations. The motivation behind this lies in the fact that many recently published studies on implicit neural representation employing rectified linear unit (ReLU)-based MLPs are incapable of capturing high-frequency details of the input signal. There are two possible explanations for this phenomenon. 1) Conventional neural network architectures encounter difficulties while learning to apply the same function at two different coordinates, and thus, the learned functions are not shift-invariant in general. 2) ReLU nonlinearities cannot parameterize any signal that has information in its second derivative since its second derivative will be zero everywhere. Therefore, the authors suggested replacing conventional nonlinearities, such as tanh or ReLU, with a periodic sine activation function to improve final results. This replacement results in gaining a certain degree of shift-invariance and also addresses the problem of the second derivative since the derivative of sine is a shifted sine itself. The method was applied to a wide variety of areas, including image, audio, and video representations, 3-D reconstruction, and solving first- and second-order differential equations. In the 3-D shape reconstruction task, SIREN generates details of complex objects and scenes better than ReLU-based implicit representations, such as NeRF [57].

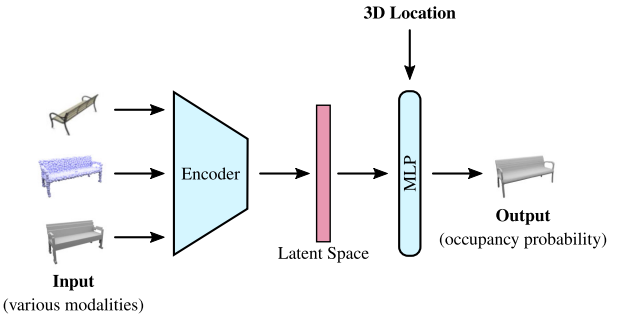


Fig. 19. *Occupancy networks architecture [45] predicts occupancy function for each point in 3-D space using a DNN. Different encoder architectures are used in occupancy networks depending on the task and input. A ResNet-18 architecture [173] for image input, a PointNet encoder [148] for point cloud input, and a 3-D-CNN for voxel input are employed.*

a) Methods based on unsigned distances: Some studies exploit unsigned distances instead of occupancy or signed distances for learning representations. With sign agnostic learning (SAL), Atzmon and Lipman [52] proposed a DL approach based on raw input data without any oriented normals or signs. Generally, regression-based methods utilize regression loss for training and need inside/outside ground-truth information for this process, such as DeepSDF [50] or occupancy networks [45]. In contrast to these methods, SAL uses a sign agnostic loss function that can be directly applied to raw unsigned data. The algorithm generates high-quality surfaces in comparison to AtlasNet [34] and a baseline method that approximates SDF based on the work by [9]. The D-Faust dataset, which comprises raw scans of humans in various poses, is used for the experiments. Although there is no need to include the signed implicit ground-truth representation in the calculation of the loss function during training and also closing surfaces for training data is unnecessary in this work, SAL predicts SDF as the final output, which also results in closing the gaps even in open surfaces and generating only closed outputs (closed surfaces, in this case, are a division of 3-D space into three regions: inside, outside, and on the surface of an object, and they do not have separate parts). Neural distance field (NDF) [53] is a method to predict the unsigned distance field for 3-D surfaces using a neural network. Similar to SAL, NDF does not close shapes during training. However, it can successfully generate open surfaces, shapes with inner structures, and open manifolds compared to IF-Net [56] and SAL [52].

DUDE [54] is another approach, which is able to represent a surface by combining the unsigned distance field

with the normal vector field. Evaluation of this method in comparison to DeepSDF and SAL demonstrates its superiority in producing high-quality outputs, especially for open surfaces, with visually pleasant renderings. The main difference between NDF and DUDE compared to SAL is that the first two can reconstruct both open and closed shapes with complex and detailed topology, while the latter attempts to close parts that should be open.

b) Part-based approaches: Encoding an entire surface into a single latent vector can lead to substantial information loss since the limited size and capacity of the latent representation causes accuracy and generalization issues [48]. In order to solve the difficulties of generalizing to other shape categories and scaling to large scenes, researchers resort to conditioning an implicit neural representation on local geometric features [46], [47], [48], [55], [56], [183], [184]. There are different approaches to the implementation of such conditioning. Some approaches fuse the volumetric representation (voxel grids) with the implicit neural representation and use local features stored in voxels for inferring implicit neural representation [46], [47], [55], [56]. Others use local patches to learn implicit neural functions [48], [183], [184]. All of these methods leveraged the advantages of encapsulating local and global information for proposing more generalizable and scalable approaches.

Jiang et al. [55] suggested the local implicit grid (LIG) representation, which decomposes 3-D space into a regular grid of overlapping part-sized local regions and encodes each region with implicit feature vectors. The key idea behind the algorithm is that objects in different categories share similar geometric features and details at neither microscale, i.e., a very small patch, nor macroscale, i.e., the entire object, but part scale. Therefore, a part-autoencoder was used to learn embeddings for different parts of an object and extract meaningful abstraction of its shape. The autoencoder consists of a 3-D-CNN encoder and an implicit network decoder in the form of a reduced version of the IM-NET [49] decoder. During inference, a pretrained implicit function decoder is used in each grid cell in order to generate the respective scene part. Eventually, the overlapping latent grids were optimized via the proposed mechanism to reconstruct the entire scene. Since this method generalizes shape priors learned from object datasets, it does not need any training on the scene-level dataset for reconstructing scenes from sparse oriented point samples. Therefore, it generates higher quality outputs from unseen object categories than other methods, such as IM-NET [49], since IM-NET learns only a single embedding for an entire object. Compared to traditional surface reconstruction methods such as PSR [8], [9], LIG is capable of recovering thin structures and details very well.

Likewise, Chibane et al. [56] introduced implicit feature networks (IF-Nets) that are composed of an encoding and a decoding tandem. The network takes voxels or point clouds as the input and predicts whether point p lies inside or outside of an object, resulting in a continuous surface at arbitrary resolution. To encode local and global structures of a 3-D shape, a 3-D multiscale grid of deep features is extracted instead of using a single vector to summarize an entire object. Consequently, rather than classifying (x, y, z) point coordinates directly, the decoder classifies a point based on these extracted features and creates occupancy predictions. IF-NET achieves better quantitative results than occupancy networks [45], point set generation network [30], deep MC [42], and IM-NET [49] in point cloud completion, voxel super-resolution, and single-view human reconstruction tasks. Moreover, Chibane and Pons-Moll [185] proposed an extension of IF-Nets for 3-D texture completion.

Peng et al. [46] developed convolutional occupancy networks, a hybrid voxel grid/implicit neural representation-based approach that combines convolution operations with implicit representations in the form of a convolutional encoder with an implicit occupancy decoder. The method is independent of the input representation. Given a point cloud or voxel grid as input, the method uses a 2-D plane encoder/3-D volume encoder based on PointNet to process the input by converting it into features and projecting these local features onto a plane(s)/volume. A convolutional 2-D plane decoder/3-D volume decoder further processes the feature plane(s)/volume using 2-D/3-D U-Nets [186], [187], integrating both local and global information. In the end, a small fc occupancy networks [45] is used to predict the occupancy probability from a given query point p and its feature in 3-D space. For rendering and extracting meshes from the input, the MISE algorithm is applied during inference. Evaluation of both object- and scene-level reconstructions was performed using synthetic and real-world datasets. The major difference between the novel method [46] and the original occupancy networks [45] is that convolutional occupancy networks capture the local features of the space and global features, leading to higher generalizability, scalability, and faster training. Moreover, it benefits from the translational equivariance property of convolutional networks while not supporting the rotational equivariance property.

In a similar work, Chabra et al. [47] introduced deep local shapes (DeepLSs), a method for deep shape representation, which uses learned local shape priors. As illustrated in Fig. 20, the key idea is the decomposition of a shape into small components in order to improve reconstruction results. To this end, local information of these components is stored in a grid of independent latent codes. Based on these, SDFs are predicted by applying DeepSDF [50] as a local shape neural network to each grid cell. DeepLS outperforms DeepSDF in accuracy and inference time by approximately an order of magnitude.

Unlike occupancy networks [45] and DeepSDF [50], which extract the global latent code vector from the entire input, local patches are modeled as deep implicit functions in patch-based approaches [48], [183], [184]. Erler et al. [48] presented a patch-based learning framework, called Points2Surf, which generates accurate implicit

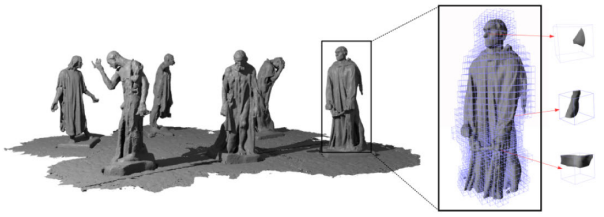


Fig. 20. *DeepLS [47] decomposes a scene into local shapes and uses a set of locally learned continuous SDFs defined by a neural network.*

surfaces directly from raw point clouds without surface normals. The underlying algorithm is based on the notion of considering a shape as a collection of small shape patches. Instead of representing an entire surface as a single latent vector, Points2Surf creates separate feature vectors for different patches to describe local details in addition to global information. By decomposing the surface reconstruction problem into learning a global function (that learns the sign of SDF) and a local function (that learns the absolute distance field of SDF with respect to local patches), Points2Surf succeeds in being robust to noise and missing parts and also generalizing well to unseen shapes. In addition, Points2Surf yields a significant drop in the reconstruction error on unseen classes compared to both data- and nondata-driven methods, such as DeepSDF [50] and AtlasNet [34], or SPR [9]. However, this patch-based approach results in longer computation time, inconsistencies between outputs of neighboring patches, and nonwatertight and bumpy surfaces.

There are a growing number of studies based on implicit neural representation for various tasks. Some authors investigated 3-D human reconstruction [184], [188], [189], 4-D reconstruction [190], and 3-D reconstruction of the appearance and texture of surfaces in addition to their geometry with 3-D supervision [191], [192] or without 3-D supervision [57], [193], [194], [195]. These are some recent articles [196], [197], [198], [199], [200], [201], which are SDF-based, and some [184], [188], [189], [202] that are based on predicting occupancy probability.

c) Equivariant neural networks: Chatzipantazis et al. [203] introduced an SE(3)-equivariant coordinate-based attention network called TF-ONet for 3-D surface reconstruction. Local shape modeling and equivariance are the two core design principles of this method. SE(3) stands for special Euclidean group in three dimensions representing transformations including translations and rotations in 3-D. In simple terms, equivariance means that, when the pattern in the input changes, i.e., when it is rotated or shifted to a specific direction, the output should also change in an equivalent proportion. TF-ONet works directly on unoriented and irregular point clouds and outputs the occupancy field of a shape. To predict the occupancy score at any given point in space, TF-ONet creates equivariant features for each point that function as keys and values of specialized attention blocks. This enables TF-ONet to output high-quality reconstructions and generalize to novel scenes composed of multiple objects, despite being trained on single objects in canonical poses. Inspired by SE(3) transformers [204] and tensor field networks [205], TF-ONet attention modules ensure equivariance by incorporating symmetries into the learning process. It is basically a two-level approach. 1) The first level, i.e., an encoder, applies self-attention in local neighborhoods around each point to infer local features from the point cloud. 2) The second level, i.e., a cross-attention occupancy network, uses the extracted point features and the coordinates of a query point in space to calculate the value of the occupancy function for the specific query point.

For single-object reconstruction tasks, TF-Onet performs comparably better than nonequivariant networks, such as occupancy networks [45], convolutional occupancy networks [46], IF-Net [56], and also equivariant networks, such as vector neurons [206] and GraphOnet [207] considering evaluation metrics, such as Chamfer-L1, F1-score, and IoU. For scene reconstruction tasks trained only on single objects, global shape modeling-based techniques, such as occupancy networks [45] and vector neurons [206], are not able to generalize to scenes containing multiple objects. Moreover, local shape modeling-based methods, such as convolutional occupancy networks [46], which are not equivariant under SE(3) transforms, are only able to produce low-quality objects in novel poses. TF-ONet instead excels at the tasks and can generalize to novel scenes with high quality, benefiting both from local shape modeling and equivariant properties.

2) NeRF-Based Approaches:

a) Fundamentals of NeRF: NeRFs [57], commonly referred to as NeRFs, are basically used for view synthesis. The main idea behind NeRFs is to train a model that can produce new views of a scene or an object and can represent them in 3-D, given a set of 2-D images from different viewing angles as input. Hence, multiple input views of a scene and their corresponding camera poses are used to render new views of that scene by interpolating between the given views. The NeRF method employs an fc deep network to represent a scene. Each input (x, y, z, θ, ϕ) is a single continuous 5-D coordinate that encompasses spatial position and viewing direction, and each output $(\text{RGB}\sigma)$ is density and view-dependent emitted radiance at that particular spatial location. Consequently, the neural network describes an implicit function that exists throughout all locations as a continuous representation without any discretization. As a result, by implicitly encoding density and color through a neural network, NeRF has demonstrated impressive performance on new view synthesis of a particular scene.

Although overfitting is usually an undesirable behavior in machine learning, the key part of this approach is the usage of a neural network that is overfitted to one

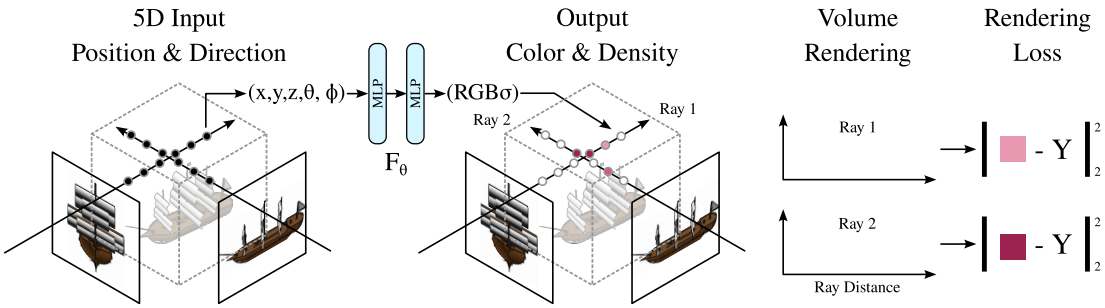


Fig. 21. *Overview of NeRFs [57]. The ship in the figure is borrowed from the ShapeNet dataset [96].*

particular scene and only cares about this specific scene. For rendering a new scene, it is necessary to take a fresh neural network and train it from scratch until it is overfitted to the new scene. Therefore, instead of storing a scene as a mesh or a voxel grid, the scene is stored in the weights of the neural network. For instance, if a scene consists of a tree, the weights represent this tree and are very specific to it, outputting nonsense for another scene if not being trained once again.

To explain the fundamentals of NeRF in more detail according to Fig. 21, the images have to be transformed to 5-D coordinates (x, y, z, θ, ϕ)s first. (x, y, z) are coordinates of a pixel point in 3-D space, and (θ, ϕ) are related to the viewing angle. For each pixel on an image, a ray is sent through. Therefore, every pixel in every input picture defines a ray, and then, it is sampled along the ray. Consequently, each input image sends out a lot of rays, and for each ray, there are many sampled points. Next, for each location represented as (x, y, z, θ, ϕ), the neural network effectively determines the presence of an object and subsequently identifies its corresponding color. This nine-layer fc network provides four numbers (RGBσ) as the output: the (RGB) is the color of that particular pixel point, and σ is the density for each of the individual points. The density value serves as an indicator of the presence or absence of an object in the designated region of space, as well as its density. If this process is done for all the points in space from all viewing angles, a complete 3-D representation of what it looks like can be inferred. The neural network outputs different results for the same location depending on different viewing angles. Accordingly, it can capture the reflections, lighting effects, and transparency. Eventually, classical techniques for volume rendering are employed to project the network outputs onto a 2-D image. Given that volume rendering is intrinsically differentiable, it is possible to define a loss function that measures the difference between the predicted and the ground-truth color of the ray. In order to convert NeRF to a mesh, MC can be further applied.

To produce high-resolution complex scenes, two interesting tricks were utilized: 1) positional encoding; 2) hierarchical sampling. Positional encoding, which is similar to the same one in transformers [208], is used to map the 5-D input vector to higher dimensional space using sin and cos waves, helping MLP in approximating and representing high-frequency functions. It enhances the ability of a neural network to not only capture coarse-grained structures but also to perform well in representing finer details. Hierarchical sampling is a two-step sampling method with two networks: a coarse network and a fine network. The points on the ray are sampled in a uniformly distant fashion from each other. These sampled points are run through the network for density prediction. Next, an evaluation step is taken place to decide where should be sampled more in the second round, based on the output of the previous step. Thus, the output of the coarse network discloses where the important stuff is. The second round of sampling starts with points with higher density, i.e., points closer to the particular object that is perceived, and the vicinity of such points will be sampled a lot more. Both coarse- and fine-grained networks are optimized at the same time using a loss.

Delving into the advantages associated with NeRFs, it is clear that these methods are not view-dependent, without the need for any 3-D input supervision. In addition, NeRFs are memory-efficient compared to voxel grid representation. One neural network of one scene fits into a few megabytes, which might even be smaller than the input image size for that scene, whereas dozens of gigabytes might be needed for storing the same scene in voxels. Regarding the limitations of NeRFs, what makes them impractical is their requirement for a large number of high-quality posed images as input. The more images are fed, the better the output quality will be. Another downside is related to their high computational cost, originating from optimizing each scene individually without sharing knowledge between different scenes [62]. This implies that, for every scene, the network should be trained again, and a pretrained one cannot be utilized. For instance, it takes around 100k–300k iterations, i.e., roughly one to two days, for the naive NeRF network [57] to be trained on a single scene using a single NVIDIA V100 GPU.

b) NeRF and its variants for view synthesis: This section provides a summary of some of the papers that aim to enhance NeRF and its abilities. In NeRF++, Zhang et al. [61] analyzed NeRF and uncovered three major problems and situations in which NeRF might fail: shape-radiance ambiguity, near-field ambiguity, and

parameterization of unbounded scenes, such as large real-world scenes. The first two issues are related to the fact that NeRF is actually overparameterized, i.e., the degree of freedom for NeRF to hallucinate and move toward a completely wrong answer is high. However, the authors of NeRF [57] use an interesting implementation trick and regularization. They feed viewing angles in the very last layers of the MLP network. Therefore, the MLP actually starts with locationwise coordinates of a point in the beginning, and viewing angles are fed in the last layers, resulting in a limited degree of freedom for NeRF. Accordingly, if all 5-D coordinates are fed to the network from the beginning, the shape radiance ambiguity becomes a big issue, affecting the quality of NeRF's outputs drastically.

NeRF++ proposes a couple of solutions to tackle these three problems and enhance output quality. By introducing an auxiliary loss, NeRF can avoid moving toward a poor solution, which may lead to completely wrong scene geometry estimation, thus addressing the shape-radiance ambiguity issue. Furthermore, adaptive near-field culling is proposed to solve the near-field ambiguity issue. It culls the front part of each view frustum adaptively based on the geometry of a scene, i.e., it prevents estimating the geometry right in front of the camera contrary to vanilla NeRF. The third issue concerns scenarios in real-world settings where precise reconstruction of objects in front of the camera is essential. However, the camera's ability to capture other items beyond these objects necessitates a certain level of reconstruction for the distant items as well. NeRF++ suggests homogenous parameterization that enables having a detailed reconstruction in the foreground and a detailed reconstruction of the background. This is done by training two NeRFs, one for the foreground part of the scene and the other for the background part, increasing the capacity of the model for reconstructing details at different levels. NeRF++ still needs per-scene training, and one scene takes about three days to be trained.

PixelNeRF [62] is built upon the concept of NeRFs for 3-D reconstruction and synthesizing photorealistic 3-D scenes from a single or a small number of posed images. PixelNeRF attempts to tackle the requirement of NeRFs for a lot of images as the input and make it generalizable. Considering the fact that extracting 3-D geometry and the appearance of a scene from limited input is a challenging task, and NeRFs do not share knowledge between the scenes, the framework proposes to condition an NeRF on spatial image features. Thus, pixelNeRF employs a fully convolutional image encoder that infers a pixel-aligned feature grid. Then, a spatial location and its corresponding encoded feature are fed to an NeRF network for color and density prediction. PixelNeRF shows better generalization capabilities and performance compared to NeRF. However, its rendering time is still slow, and more input views cause a linear increase in the runtime.

In another concurrent work to overcome the generalizability issue and long optimization time of NeRFs, MVSNeRF [63] suggests a DNN that can reconstruct an NeRF, given only three nearby input views. This approach combines plane-swept cost volumes, which are used for geometry-aware scene reasoning in MVS, with NeRF models. To create a cost volume, MVSNeRF first warps 2-D image features onto a plane sweep. Then, a 3-D-CNN is leveraged for the reconstruction of a neural encoding volume with per-voxel neural features. Next, features interpolated from the encoding volume are employed to predict density and RGB radiance for an arbitrary point using an MLP. Achieving comparable or better rendering results, MVSNeRF can significantly surpass NeRFs [57] in terms of optimization time efficiency, i.e., roughly 30 times faster, if more images are provided as input. Moreover, it generalizes better than PixelNeRF [62] and IBRNet [209].

MipNeRF [210] attempts to address one of the problems of NeRF, which is the production of blurred or aliased renderings when dealing with training or testing images at different scales. In NeRF, all of the cameras have the same distance from an object. Thus, it is able to do view synthesis without the need to think about scaling or aliasing. However, when new cameras are to be added at different scales, NeRF begins to collapse since it is a single-scale model trying to tackle a multiscale problem. To fix this issue, MipNeRF proposes some modifications to the vanilla NeRF including the following: 1) casting a cone instead of sending a ray through each pixel; 2) slicing up the cone into conical frustums instead of sampling single points along each ray; 3) computing integrated positional encoding instead of positional encoding of a single coordinate along the ray; 4) in general, training a single neural network that describes the scene at multiple scales instead of training separate neural networks at various scales. These new properties help MipNeRF reason about the scale of its inputs. MipNeRF is capable of producing high-resolution renderings across multiple scales rather than just at a single scale in vanilla NeRF. NeRF's performance decreases when being trained on multiscale data, while MipNeRF's performance does not. The number of parameters in MiPNeRF is half of that in NeRF while also being 7% faster for their multiscale dataset. Mip-NeRF360 [211] and ZipNeRF [212] are some other recent methods used for antialiasing NeRFs.

In a work proposed by NVIDIA, instant NGP, Müller et al. [213] try to facilitate and speed up neural graphics primitive tasks. A neural graphics primitive is an object represented by a neural network that takes a query as input, such as position and some extra parameters, and outputs appearance and shape attributes. Examples of NGP can be computing SDF, NeRFs, radiance cashing, and so on. To bring about simplicity, instant training, real-time rendering, and high-quality results for instant NGP, solutions such as multiresolution hash encoding by storing the trainable feature vectors in a compact spatial hash table, using a small neural network called a fully fused neural network, and improvement of

Table 2 Comparison of Various 3-D Reconstruction Methods: CNN, GCN, IoU, CD, EMD, HD, Binary Cross-Entropy (BCE), AP, Cross-Entropy (CE), Squared Distance Error (SDE), NC, Peak Signal-to-Noise Ratio (PSNR), and Structural Similarity Index (SSIM)

Name	Input	Output	Method	Loss	Dataset	Metric
3D-R2N2 [20]	A single-view image or multi-view images	Voxels	Encoder-decoder, 3D conv LSTM	Voxel-wise CE	ShapeNet, Pascal 3D [229], Online products [230]	IoU, CE
VRN [21]	Voxels	Voxels	Encoder-decoder	BCE	ModelNet	Accuracy
TL-embedding [22]	RGB images	Voxels	Encoder-decoder	CE, Euclidean loss	ShapeNet, IKEA dataset [231]	AP
3D-GAN [23]	An image	Voxels	CNN, 3D GAN, Encoder-decoder	BCE loss, KL-divergence loss, reconstruction loss	ModelNet, IKEA dataset, ShapeNet	AP
3D-EPN [24]	Depth maps	Voxels	Encoder-predictor network	L1 loss	ShapeNet	Accuracy, L1 error
3D shape completion [25]	Point cloud and a 3D bounding box	Occupancy grid or SDFs	Encoder-decoder	Reconstruction loss, maximum likelihood loss	ShapeNet, KITTI, ModelNet	Hamming distance, accuracy, completeness
SG-NN [139]	RGB-D scan	A sparse TSDF	Encoder-decoder	L1 loss, BCE loss	Matterport3D	L1 error
Octnetfusion [26]	One/multiple 2.5D depth image(s)	Voxel or Octree	Encoder-decoder	L1 loss, BCE	ModelNet40	IoU, precision, recall
HSP [28]	RGB images or depth images or partial grids	Voxels	Encoder-decoder	CE	ShapeNet	IoU, CD
OGN [29]	Voxels	Structure of an octree and binary occupancy map	Encoder-decoder	CE	ShapeNet	IoU
Adaptive O-CNN [27]	A single image or point cloud	A patch-guided adaptive octree	Encoder-decoder	CE, SDE	ModelNet, ShapeNet	CD, accuracy
Point set generation net [30]	A single RGB or RGB-D image	Point cloud	Encoder-predictor	CD, EMD	ShapeNet	IoU, CD, EMD
Latent 3D points [31]	Point cloud	Point cloud	Encoder-decoder, GAN	CD, EMD	ShapeNet, ModelNet	JSD, coverage, MMD
FoldingNet [32]	Point cloud	Point cloud	Encoder(graph-based)-decoder	CD	ShapeNet, ModelNet	Accuracy
PointFlow [33]	Point cloud	Point cloud	Encoder-decoder	Prior loss, reconstruction loss, posterior loss	ShapeNet	JSD, MMD, coverage, CD, EMD, accuracy
AtlasNet [34]	2D images or point cloud	Mesh	Encoder-decoder	CD loss	ShapeNet	CD
Meshlet [37]	Point cloud	Mesh	Encoder-decoder	CD loss	ShapeNet	CD, HD
Pixel2Mesh [38]	An RGB image	Mesh	Graph convolution network	CD loss, normal loss	Dataset of 3D-R2N2 [20]	F1-score, CD, EMD
Pixel2Mesh++ [39]	A few RGB images or multi-view images	Mesh	GCN	CD loss, normal loss	Dataset of 3D-R2N2 [20]	F1-score, CD
CMR [40]	An image	Mesh	Convolutional encoder	Reprojection loss, regression loss	CUB-200-2011 dataset, PASCAL 3D+ dataset	IoU
Point2Mesh [41]	Point cloud	Mesh	CNN	CD loss, beam-gap loss	A large dataset of object scans [232]	F1-score
DMC [42]	Point cloud	Mesh	Encoder-decoder network with skip connections	Point to mesh loss, occupancy loss, smoothness loss, curvature loss	ShapeNet	CD, accuracy, completeness
Scan2Mesh [43]	One/multiple depth image(s)	Mesh	CNN, graph neural network	CE, CD loss	ShapeNet	CD, normal deviation

Table 2 *(Continued.)* Comparison of Various 3-D Reconstruction Methods: CNN, GCN, IoU, CD, EMD, HD, Binary Cross-Entropy (BCE), AP, Cross-Entropy (CE), Squared Distance Error (SDE), NC, Peak Signal-to-Noise Ratio (PSNR), and Structural Similarity Index (SSIM)

Mesh R-CNN [44]	An RGB image	A category label, segmentation mask, boundary box, a 3D triangular mesh	GCN	BCE, CD	ShapeNet, Pix3D dataset	F1-score, CD, NC
Meshing point clouds with IER [35]	Point cloud	Mesh	CNN	Euclidean distance, geodesic distance	ShapeNet	F1-score, CD, NC
REIN [36]	Point cloud	Mesh	Encoder-decoder, RNN	CD, BCE	ShapeNet, ModelNet10	CD, point normal similarity
Occupancy Nets [45]	An image or point cloud or discrete voxel grids	Implicit surface	Encoder, fully connected network	CE	ShapeNet, KITTI, Pix3D	IoU, CD, NC
IM-Net [49]	Images or voxels	Implicit surface	Encoder-decoder, GAN	Weighted mean squared error, Wasserstein GAN loss	ShapeNet	MSE, IoU, CD, LFD, MMD, COV
DeepSDF [50]	Point cloud	Implicit surface	Auto-decoder	L1 loss	ShapeNet	CD, EMD, accuracy
SIREN [51]	Point cloud	Implicit surface	Fully connected neural network	SDF loss (Eikonal equation)	Stanford 3D scanning repository	N/A
SAL [52]	Point cloud or triangle soups	Implicit surface	Variational encoder-decoder	Sign-agnostic loss with L2 distance	D-Faust dataset	CD
NDF [53]	Point cloud	Implicit surface	Encoder-decoder	Unsigned distance field loss	ShapeNet	CD
DUDE [54]	Triangle soups	Implicit surface	Feed-forward networks	L2 loss	ShapeNet	IoU, mean absolute error, normal map error
LIG [55]	Point cloud	Implicit surface	Encoder-decoder	BCE loss	ShapeNet, Matterport 3D, SceneNet	F1-Score, CD
IF-NET [56]	Point cloud or occupancy grid	Implicit surface	Encoder-decoder	CE	ShapeNet	IoU, CD, NC
Conv occupancy nets [46]	Point cloud or voxels/coarse occupancy grid	Implicit surface	Encoder-decoder	BCE	ShapeNet, ScanNet, Matterport 3D	F1-score, IoU, CD, NC
DeepLS [47]	Depth data or mesh	Implicit surface	Autodecoder network	Negative log likelihood loss	Stanford 3D scanning repository, 3D warehouse [233]	CD
Point2surf [48]	Point cloud	Implicit surface	Encoder-decoder	L2 loss, BCE loss	ABC dataset	CD
UNISURF [223]	RGB images	Implicit surface	MLP	Reconstruction loss, Surface regularization	DTU [234], BlendedMVS [235], SceneNet	CD
NeuS [222]	RGB images	Implicit surface	MLP	Color loss, regularization loss, mask loss	DTU [234], BlendedMVS [237]	CD, PSNR, SSIM

training and rendering algorithm are proposed as main ideas.

The amount of research efforts based on NeRF is increasing. From relighting [64], [65], [214], [215] and view synthesis without pose supervision [216] to learning nonrigid objects and dynamic scenes [66], [67], [68], [217], [218], [219] and tackling computational challenges of NeRF and heading toward the real-time rendering [58], [59], [60], [220], numerous studies have been conducted to broaden the horizons of NeRF and its various applications.

c) NeRF for 3-D surface reconstruction: In an NeRF model, the scene geometry is hidden inside the neural networks, i.e., it is implicit. In order to achieve 3-D surface reconstruction and transform the NeRF representation into an explicit representation, such as a mesh, a surface extraction step is essential. By analyzing and thresholding the learned density, i.e., extracting an arbitrary level set of the density function that is learned by NeRF, and using methods such as MC, the baseline NeRF can extract and reconstruct an approximate explicit 3-D geometry [221].

Table 3 Quantitative Report About Some of the Methods' Performance on ShapeNet. CD, IoU, AP, and F_Score Are Calculated as the Average. For IoU, F_Score, and AP, the Higher the Better. For CD, the Lower the Better. The Number of ShapeNet Categories Used in an Experiment (#Cats), Not Measured or Not Mentioned (-), SVR, MVR, Reconstruction (R), Completion (C), Autoencoding (AE), Training Time (T), Inference Time (I), Generating a Mesh (mg), Memory (Mem.), and MS. * Is Calculated for (32^3). + Is Related to Chamfer-L1. For Detailed Information Regarding Data Preparation Methods, Train/Test Splits, Metrics, and Other Specific Details, Please Refer to the Context of Each Individual Paper

Papers	ShapeNet					Task	Time	Size
	CD	IoU	F_score	AP	#Cats			
3D-R2N2 [20]	-	>0.60	-	-	13	MVR (3 views)	-	-
TL-embedding [22]	-	-	-	65.40	5	SVR	-	-
OGN [29]	-	0.59 *	-	-	13	SVR	5 days T for 256^3	0.54G(256^3)
Adaptive O-CNN [27]	0.00460	-	-	-	13	SVR	-	-
Point set generation [30]	0.25000	0.64	-	-	13	SVR	-	-
PointFlow [33]	0.00070	-	-	-	13	AE	-	-
AtlasNet [34]	0.00150	-	-	-	13	AE (25 patches)	-	-
AtlasNet [34]	0.00510	-	-	-	13	SVR (25 patches)	-	-
Meshlet [37]	0.00900	-	-	-	-	R	-	-
Pixel2Mesh [38]	0.59100	-	59.72	-	13	SVR	72h T - 15.5ms I (mg)	-
Pixel2Mesh++ [39]	0.48000	-	66.48	-	13	MVR (3 views)	96h T - 15.5ms I (mg)	-
Scan2Mesh [43]	0.00160	-	-	-	8	C	-	-
Meshing with IER [35]	0.00071	-	87.20	-	8	R	<10s I/a pc with 12,800 pts	-
IM-Net [49]	0.00140	-	-	-	5	SVR	-	-
IM-Net [49]	0.00060	0.75	-	-	5	AE	-	-
DeepSDF [50]	0.00030	-	-	-	5	AE	9.72s I	0.0074(MS)
DeepSDF [50]	0.00160	-	-	-	3	C	9.72s I	0.0074(MS)
IF-NET [56]	0.00002	0.88	-	-	13	R	-	-
Occupancy Nets [45]	0.21500+	0.57	-	-	13	SVR	3s I/per mesh	-
Occupancy Nets [45]	0.07900+	0.77	-	-	13	C	3s I/per mesh	-
Conv onets [46]	0.04800+*	0.87	93.30	-	13	R	-	5.9G Mem.

Although NeRF and its variants generate impressive results for the novel view synthesis task, they cannot output high-quality 3-D surface reconstruction. The quality of the extracted 3-D geometry is not satisfactory because the initial objective of NeRF is novel view synthesis, not 3-D surface reconstruction. Since the density-based representation used in NeRFs is flexible and does not have enough constraints on 3-D geometry [222], it imposes some limitations on inferring accurate surface geometry, especially when ambiguities exist. Therefore, the extracted surfaces usually contain artifacts. To alleviate this issue, some papers have been presented for the 3-D surface reconstruction task that tried to incorporate implicit neural surface representation approaches based on an SDF or an occupancy function into NeRF-based methods, benefiting from the advantages of both categories. In these methods, instead of choosing the density-based scene representation used in NeRF, the scene space is usually represented as an SDF or an occupancy function.

Oechsle et al. [223] proposed UNIfied Neural Implicit SUrface and Radiance Fields (UNISURFs), which is a framework for 3-D surface reconstruction and capturing high-quality implicit surface geometry from multiview images without the need for object masks. It unifies the implicit surface models with radiance fields for solid and nontransparent object reconstruction given a set of RGB images. UNISURF represents surfaces and defines object or scene geometries using occupancy values. It learns and optimizes this implicit surface via a volume rendering method like NeRF. The output mesh is extracted using the MISE algorithm [45]. Considering reconstruction quality, UNISURF outperforms NeRF [57]. There are some limiting factors for this method, including reconstructing only solid objects and constraints to model transparencies, performance drop for overexposed or rarely visible regions in the ground-truth images, and inability to resolve the shape-appearance ambiguities, such as shadows and holes in objects.

In another concurrent attempt, Wang et al. [222] presented NeuS that learns neural implicit surface representation based on SDF using volume rendering, with the goal of reconstructing the 3-D surface of an object or scene given multiple images from different viewing points without leveraging mask supervision. Instead of just doing standard volume rendering or standard surface rendering, this framework suggests using volume rendering (inspired by NeRF) in addition to surface representation with neural SDF. The key idea behind this method is to represent a 3-D surface as the zero-level set of an SDF, i.e., representing a surface with neural implicit SDFs, and to introduce a new volume rendering method by taking inspiration from NeRF, for training a neural SDF representation with robustness. This novel volume rendering technique attempts to learn the weights of the neural network by rendering images from the implicit SDF first and then minimizing the difference between the rendered images and the input images. NeuS performs quantitatively and qualitatively better than NeRF [57] and UNISURF [223] in high-quality surface reconstruction. However, one failure case of NeuS is its inability to accurately reconstruct textureless regions. This limitation is caused by the ambiguity of these textureless regions for reconstruction in neural rendering.

Variants of NeuS [224], [225] have been proposed with the goal of improving the reconstruction quality. HF-NeuS [224], a method for multiview surface reconstruction with high-frequency details, breaks down the SDF

into fundamental components, namely, base and displacement functions, and adopts a gradual increase in high-frequency details through a coarse-to-fine strategy. In Geo-Neus [225], by utilizing sparse 3-D points in SfM constraint in conjunction with the photometric consistency in MVS constraint, the learning of neural SDF can be enhanced.

In a similar fashion to NeuS, another concurrent work called VolSDF [226] suggested a volume rendering framework for implicit neural surfaces. Replacing general-purpose MLP densities with densities from a certain family, i.e., in this case representing the density as a function of the signed distance to the scene's surface, is the core contribution of VolSDF. Two fc neural networks, one for the approximation of the SDF of the learned geometry and the other for representing the scene's radiance field, form the structure of this framework. Compared to NeRF [57] and NeRF++ [61], VolSDF generates more accurate results. One of the limitations of VolSDF is that it assumes the object is homogeneous with a constant density. Moreover, its reconstruction time is still high due to the independent training of the network for each scene.

Recently, SDFStudio [227], which is a framework for neural implicit surface reconstruction, has been released. It is built on top of nerfstudio [228] and includes a unified implementation of VolSDF, NeuS, and UNISURF, three popular neural implicit surface reconstruction techniques. Because of the unified and modular implementation of this framework, transferring ideas between methods is simple. The idea from Geo-NeuS can be integrated with VolSDF, bringing about the Geo-VolSDF method.

VII. DISCUSSION AND FUTURE TRENDS

In Section VI, the latest attempts toward 3-D reconstruction using DL techniques were reviewed. A summary and comparison of presented learning-based surface reconstruction approaches can be found in Table 2. Furthermore, Table 3 contains a quantitative report about the performance of some of the approaches on the ShapeNet dataset. There is a qualitative gap between 3-D models created by learning-based approaches and artist-created CAD models [43], and there are still open problems in this scope. Some of these challenges are listed in the following.

In the existing approaches, serious bottlenecks are caused by computation time and generalization power. The requirement of long training time is a drawback to the adoption of some of the DL-based approaches. On the other hand, there are concerns raised about the environmental impact of prolonged training periods. To this end, designing models with a reduced number of parameters, less complexity, and yet high performance can be a target to hit. In addition, the utilization of transfer learning may serve as a partial solution. Regarding the generalizability issue, methods with the capability of multicategory generalization, i.e., generalizing well to other topology categories, should be further investigated. One solution might be to learn latent shape spaces that are not class-specific. Consequently, as a future direction, moving toward models with comparable shorter training time and stronger generalizability can be an interesting yet reasonable strategy.

Current methods are highly dependent on an external supervisor for annotating input data. Reducing the need for supervision is a desirable trait for a learning-based approach [40]. Furthermore, there are various large-scale datasets appropriate for geometric DL tasks. However, there is still a need for creating datasets with richer 3-D annotations that are suitable for shape and surface reconstruction.

On the other hand, some of the current evaluation metrics fall short of capturing surface properties accurately. Therefore, it is necessary not to be limited to quantitative results but to explore qualitative results to gain a deeper understanding of surface details as well. Moreover, presenting better and more robust evaluation metrics, which are at the same time computationally efficient and less complex (in point cloud comparison, CD has quadratic complexity for instance), is another area that is essential to focus on.

In the context of volumetric methods, various challenges exist that should be tackled. Because of the discretization of data, some input information and details may partially be lost. Cubic growth in memory and computational costs with respect to resolution and poor scalability of these methods with resolution increase lead to difficulty in inferring high-resolution outputs. Considering the influence of 3-D resolution on the performance of volumetric CNNs for instance, better performance can be achieved by designing efficient volumetric CNN architectures for instance, which are able to scale to higher resolutions [128].

For point-based approaches, current methods extract a fixed and limited number of points from the point cloud dataset and feed them to their network architecture, thus affecting the output quality. Overcoming this limitation and implementing models with the ability to handle variable-length input can be ambitious yet interesting future directions.

In mesh-based approaches, it is challenging to define a loss on meshes, which is easy to optimize [34]. One of the limitations of patch-based approaches in the mesh-based representation category that affects the reconstruction of fine details is the usage of a fixed scale mesh patch [37]. A coarse-to-fine approach and extracting mesh patches at different scales might result in more precise outputs. On the other hand, generating a closed shape using patch-based methods, and recognizing and segmenting shapes using these methods are issues that still require solutions [34].

Implicit neural representations have recently gained popularity due to their performance and favorable properties. Existing isosurface extraction approaches used for extracting representations from implicit neural representations are computationally intensive and, thus, comprise a bottleneck. Furthermore, it may be worthwhile to combine sign-agnostic implicit neural approaches with

generative methods, such as GANs [52]. Moreover, NeRF-based approaches mostly suffer from high computational cost, long training time, and the inability to share knowledge between various scenes, thus being scene-specific networks. The necessity for more input images in order to have high-quality outputs should be alleviated. Improving NeRF-based methods' time and computation efficiency, their generalizability to unseen scenes, and their surface reconstruction ability can be important research questions.

In general, reducing the performance gap between synthetic and real-world data, proposing better and more representative evaluation metrics for quantifying shape reconstruction analysis results [49], conducting research in the challenging task of scene-level reconstruction, empowering proposed methods with multiscale reconstruction (coarse-to-fine manner) [48], implementing and employing methods for capturing high-frequency details with the purpose of reconstructing thin parts of a scene or object in high-quality, considering the equivariance concept for designing a neural network, and fusing different approaches mentioned in Fig. 6 in order to enjoy the benefits of them simultaneously are aspects that should not be ignored in future studies. In addition, the application of transformer architectures [208], i.e., a DL model that is based on the self-attention mechanism, seems to be promising in 3-D vision [237], [238], [239]. On the other hand, self-supervised learning [240], which is a technique for predicting unobserved or hidden part of the input from observed or not hidden part of the input, can be one of the interesting approaches for solving reconstruction and in general computer vision problems with low quality and a limited amount of data. Furthermore, considering the current interest, diffusion models [241], [242], [243], [244], which learn to infer and generate a meaningful output from pure noise, seem to be another exciting approach to be used in 3-D generation, completion, and reconstruction [245], [246].

It is equally expected that surface reconstruction applications play an increasingly important role. One of the major uses will be in observational RS-related disciplines where surface reconstruction will aid in archeological discoveries, agriculture, disaster prevention and response, and cartography. Equally, design- or projection-based applications have great utilization potential for learned surface reconstruction, including, but not limited to, 3-D modeling in games and movies, architecture, or CAD. Yet, all of the aforementioned scenarios are considering only (close to) static surfaces. The anticipation is that accurate reconstruction of dynamically changing objects and environments, nonrigid objects or scenes, textureless regions and transparent objects, and overcoming the challenges of rarely visible regions, occlusions, shadows, and holes in an object or scene will be crucial and consequential next steps in this field of study. Overall, more applications of neural learning approaches will emerge for surface reconstruction, especially in SFX and VFX animation, human reconstruction, robotics, autonomous driving, and medicine.

VIII. CONCLUSION

In this article, we provided a review of the state-of-the-art approaches for learning-based 3-D surface reconstruction. We have taken no special perspective, making the manuscript accessible not only to method researchers but also to applied users seeking to contextualize these approaches for their domains.

For this, we have reiterated commonly used open and accessible benchmarking datasets, different input and output data modalities, and some acquisition techniques. To make processing results comparable, we have highlighted widely used metrics for evaluating learned models and detailed their particularities.

The main part of this article has introduced DL-based 3-D surface reconstruction approaches. In summary, these can be classified into four major categories based on their output representations: 1) voxel-based; 2) point-based representation; 3) mesh-based; and 4) implicit neural. For each of the categories, we listed some well-known methods, explaining their contributions, challenges, strengths, and weaknesses.

Although 3-D deep surface reconstruction has made impressive progress over the last few years, there are several remaining challenges. The following nonexhaustive list will highlight the major open issues:

1) computation time;
2) generalizability;
3) energy consumption and environmental impact;
4) representation compression;
5) resolution;
6) water tightness;
7) nonrigid, dynamic, or transparent object reconstruction;
8) reconstruction of rarely visible or occluded regions, shadows, and holes in an object or a scene.

Toward the end of this article, we discussed current challenges and possible future trends in deep 3-D surface reconstruction. We assume that coming research will put a strong emphasis on self-attention-based models due to their exceling performance in DL in general and 2-D computer vision problems, i.e., vision transformer and its derivatives, in particular. Moreover, self-supervision will be the strong community focus due to its ability to not only improve reconstructive performance overall but also to leverage small and potentially domain-specific datasets. The application of diffusion models seems to be a promising direction as well. Finally, albeit in a niche setting, the quantification of reconstruction uncertainties will be of utmost importance for safety-critical applications and certain scientific application settings. ■

Acknowledgment

The authors thank their funding agencies.

REFERENCES

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep Learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[2] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[3] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Netw.*, vol. 61, pp. 85–117, Jan. 2015, doi: 10.1016/j.neunet.2014.09.003.

[4] L. Chen, S. Peng, and X. Zhou, "Towards efficient and photorealistic 3D human reconstruction: A brief survey," *Vis. Informat.*, vol. 5, no. 4, pp. 11–19, Dec. 2021, doi: 10.1016/j.visinf.2021.10.003.

[5] Y. Tian, H. Zhang, Y. Liu, and L. Wang, "Recovering 3D human mesh from monocular images: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, Jul. 26, 2023, doi: 10.1109/TPAMI.2023.3298850.

[6] M. Götz, C. Bodenstein, and M. Riedel, "HPDBSCAN," in *Proc. Workshop Mach. Learn. High-Perform. Comput. Environ.*, Nov. 2015, pp. 1–10, doi: 10.1145/2834892.2834894.

[7] M. Götz, G. Cavallaro, T. Géraud, M. Book, and M. Riedel, "Parallel computation of component trees on distributed memory machines," *IEEE Trans. Parallel Distrib. Syst.*, vol. 29, no. 11, pp. 2582–2598, Nov. 2018, doi: 10.1109/TPDS.2018.2829724.

[8] M. Kazhdan, M. Bolitho, and H. Hoppe, "Poisson surface reconstruction," in *Proc. Eurographics Symp. Geometry Process.*, vol. 7, pp. 61–70, Jun. 2006, doi: 10.2312/SGP/SGP06/061-070.

[9] M. Kazhdan and H. Hoppe, "Screened Poisson surface reconstruction," *ACM Trans. Graph.*, vol. 32, no. 3, pp. 1–13, Jun. 2013, doi: 10.1145/2487228.2487237.

[10] F. Bernardini, J. Mittleman, H. Rushmeier, C. Silva, and G. Taubin, "The ball-pivoting algorithm for surface reconstruction," *IEEE Trans. Vis. Comput. Graphics*, vol. 5, no. 4, pp. 349–359, Dec. 1999, doi: 10.1109/2945.817351.

[11] J.-D. Boissonnat and B. Geiger, "Three-dimensional reconstruction of complex shapes based on the Delaunay triangulation," in *Proc. SPIE*, vol. 1905, pp. 964–975, Jul. 1993, doi: 10.1117/12.148710.

[12] S. Fortune, "Voronoi diagrams and Delaunay triangulations," in *Handbook of Discrete and Computational Geometry*. New York, NY, USA: Taylor & Francis, 1995, ch. 27, pp. 225–265, doi: 10.1142/9789812831699_0007.

[13] E. Che, J. Jung, and M. Olsen, "Object recognition, segmentation, and classification of mobile laser scanning point clouds: A state of the art review," *Sensors*, vol. 19, no. 4, p. 810, Feb. 2019, doi: 10.3390/s19040810.

[14] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, and M. Bennamoun, "Deep learning for 3D point clouds: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 12, pp. 4338–4364, Dec. 2021, doi: 10.1109/TPAMI.2020.3005434.

[15] L. Jiao et al., "A survey of deep learning-based object detection," *IEEE Access*, vol. 7, pp. 128837–128868, 2019, doi: 10.1109/ACCESS.2019.2939201.

[16] Y. Xie, J. Tian, and X. X. Zhu, "Linking points with labels in 3D: A review of point cloud semantic segmentation," *IEEE Geosci. Remote Sens. Mag.*, vol. 8, no. 4, pp. 38–59, Dec. 2020, doi: 10.1109/MGRS.2019.2937630.

[17] M. Berger et al., "A survey of surface reconstruction from point clouds," *Comput. Graph. Forum*, vol. 36, no. 1, pp. 301–329, Jan. 2017, doi: 10.1111/cgf.12802.

[18] M. Zollhöfer et al., "State of the art on 3D reconstruction with RGB-D cameras," *Comput. Graph. Forum*, vol. 37, no. 2, pp. 625–652, May 2018, doi: 10.1111/cgf.13386.

[19] X.-F. Han, H. Laga, and M. Bennamoun, "Image-based 3D object reconstruction: State-of-the-art and trends in the deep learning era," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 5, pp. 1578–1604, May 2021, doi: 10.1109/TPAMI.2019.2954885.

[20] C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese, "3D-R2N2: A unified approach for single and multi-view 3D object reconstruction," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 628–644, doi: 10.1007/978-3-319-46484-8_38.

[21] A. Brock, T. Lim, J. M. Ritchie, and N. Weston, "Generative and discriminative voxel modeling with convolutional neural networks," 2016, *arXiv:1608.04236*.

[22] R. Girdhar, D. F. Fouhey, M. Rodriguez, and A. Gupta, "Learning a predictable and generative vector representation for objects," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 484–499, doi: 10.1007/978-3-319-46466-4_29.

[23] J. Wu, C. Zhang, T. Xue, W. T. Freeman, and J. B. Tenenbaum, "Learning a probabilistic latent space of object shapes via 3D generative-adversarial modeling," 2016, *arXiv:1610.07584*.

[24] A. Dai, C. R. Qi, and M. Nießner, "Shape completion using 3D-encoder-predictor CNNs and shape synthesis," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6545–6554, doi: 10.1109/CVPR.2017.693.

[25] D. Stutz and A. Geiger, "Learning 3D shape completion from laser scan data with weak supervision," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 1955–1964, doi: 10.1109/CVPR.2018.00209.

[26] G. Riegler, A. O. Ulusoy, H. Bischof, and A. Geiger, "OctNetFusion: Learning depth fusion from data," in *Proc. Int. Conf. 3D Vis. (3DV)*, Oct. 2017, pp. 57–66, doi: 10.1109/3DV.2017.00017.

[27] P.-S. Wang, C.-Y. Sun, Y. Liu, and X. Tong, "Adaptive O-CNN: A patch-based deep representation of 3D shapes," *ACM Trans. Graph.*, vol. 37, no. 6, pp. 1–11, Dec. 2018, doi: 10.1145/3272127.3275050.

[28] C. Häne, S. Tulsiani, and J. Malik, "Hierarchical surface prediction for 3D object reconstruction," in *Proc. Int. Conf. 3D Vis. (3DV)*, Oct. 2017, pp. 412–420, doi: 10.1109/3DV.2017.00054.

[29] M. Tatarchenko, A. Dosovitskiy, and T. Brox, "Octree generating networks: Efficient convolutional architectures for high-resolution 3D outputs," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2107–2115, doi: 10.1109/ICCV.2017.230.

[30] H. Fan, H. Su, and L. J. Guibas, "A point set generation network for 3D object reconstruction from a single image," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 605–613, doi: 10.1109/CVPR.2017.264.

[31] P. Achlioptas, O. Diamanti, I. Mitliagkas, and L. Guibas, "Learning representations and generative models for 3D point clouds," in *Proc. 35th Int. Conf. Mach. Learn.*, vol. 80, pp. 40–49, Jul. 2018. [Online]. Available: http://proceedings.mlr.press/v80/achlioptas18a.html

[32] Y. Yang, C. Feng, Y. Shen, and D. Tian, "FoldingNet: Point cloud auto-encoder via deep grid deformation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 206–215, doi: 10.1109/CVPR.2018.00029.

[33] G. Yang, X. Huang, Z. Hao, M.-Y. Liu, S. Belongie, and B. Hariharan, "PointFlow: 3D point cloud generation with continuous normalizing flows," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 4540–4549, doi: 10.1109/ICCV.2019.00464.

[34] T. Groueix, M. Fisher, V. G. Kim, B. C. Russell, and M. Aubry, "A papier-mache approach to learning 3D surface generation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 216–224, doi: 10.1109/CVPR.2018.00030.

[35] M. Liu, X. Zhang, and H. Su, "Meshing point clouds with predicted intrinsic-extrinsic ratio guidance," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*. Cham, Switzerland: Springer, 2020, pp. 68–84, doi: 10.1007/978-3-030-58598-3_5.

[36] R. Daroya, R. Atienza, and R. Cajote, "REIN: Flexible mesh generation from point clouds," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2020, pp. 352–353, doi: 10.1109/CVPRW50498.2020.00184.

[37] A. Badki, O. Gallo, J. Kautz, and P. Sen, "Meshlet priors for 3D mesh reconstruction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 2846–2855, doi: 10.1109/CVPR42600.2020.00292.

[38] N. Wang, Y. Zhang, Z. Li, Y. Fu, W. Liu, and Y.-G. Jiang, "Pixel2Mesh: Generating 3D mesh models from single RGB images," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*. Cham, Switzerland: Springer, 2018, pp. 52–67, doi: 10.1007/978-3-030-01252-6_4.

[39] C. Wen, Y. Zhang, Z. Li, and Y. Fu, "Pixel2Mesh++: Multi-view 3D mesh generation via deformation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 1042–1051, doi: 10.1109/ICCV.2019.00113.

[40] A. Kanazawa, S. Tulsiani, A. A. Efros, and J. Malik, "Learning category-specific mesh reconstruction from image collections," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 371–386, doi: 10.1007/978-3-030-01267-0_23.

[41] R. Hanocka, G. Metzer, R. Giryes, and D. Cohen-Or, "Point2Mesh: A self-prior for deformable meshes," *ACM Trans. Graph.*, vol. 39, no. 4, pp. 1–12, Aug. 2020, doi: 10.1145/3386569.3392415.

[42] Y. Liao, S. Donné, and A. Geiger, "Deep marching cubes: Learning explicit surface representations," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2916–2925, doi: 10.1109/CVPR.2018.00308.

[43] A. Dai and M. NieBner, "Scan2Mesh: From unstructured range scans to 3D meshes," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 5574–5583, doi: 10.1109/CVPR.2019.00572.

[44] G. Gkioxari, J. Johnson, and J. Malik, "Mesh R-CNN," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 9784–9794, doi: 10.1109/ICCV.2019.00988.

[45] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger, "Occupancy networks: Learning 3D reconstruction in function space," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4460–4470, doi: 10.1109/CVPR.2019.00459.

[46] S. Peng, M. Niemeyer, L. Mescheder, M. Pollefeys, and A. Geiger, "Convolutional occupancy networks," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*. Cham, Switzerland: Springer, 2020, doi: 10.1007/978-3-030-58580-8_31.

[47] R. Chabra et al., "Deep local shapes: Learning local SDF priors for detailed 3D reconstruction," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2020, pp. 608–625, doi: 10.1007/978-3-030-58526-6_36).

[48] P. Erler, P. Guerrero, S. Ohrhallinger, N. J. Mitra, and M. Wimmer, "POINTS2SURF learning implicit surfaces from point clouds," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 108–124, doi: 10.1007/978-3-030-58558-7_7.

[49] Z. Chen and H. Zhang, "Learning implicit fields for generative shape modeling," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 5939–5948, doi: 10.1109/CVPR.2019.00609.

[50] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove, "DeepSDF: Learning continuous signed distance functions for shape representation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 165–174, doi: 10.1109/CVPR.2019.00025.

[51] V. Sitzmann, J. Martel, A. Bergman, D. Lindell,

and G. Wetzstein, "Implicit neural representations with periodic activation functions," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 7462–7473. [Online]. Available: https://proceedings.neurips.cc//paper/2020/file/53c04118df112c13a8c34b38343b9c10-Paper.pdf

[52] M. Atzmon and Y. Lipman, "SAL: Sign agnostic learning of shapes from raw data," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 2562–2571, doi: 10.1109/CVPR42600.2020.00264.

[53] J. Chibane, A. Mir, and G. Pons-Moll, "Neural unsigned distance fields for implicit function learning," 2020, *arXiv:2010.13938*.

[54] R. Venkatesh, S. Sharma, A. Ghosh, L. Jeni, and M. Singh, "DUDE: Deep unsigned distance embeddings for hi-fidelity representation of complex 3D surfaces," 2020, *arXiv:2011.02570*.

[55] C. Jiang, A. Sud, A. Makadia, J. Huang, M. Nießner, and T. Funkhouser, "Local implicit grid representations for 3D scenes," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 6000–6009, doi: 10.1109/CVPR42600.2020.00604.

[56] J. Chibane, T. Alldieck, and G. Pons-Moll, "Implicit functions in feature space for 3D shape reconstruction and completion," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 6970–6981, doi: 10.1109/CVPR42600.2020.00700.

[57] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "NeRF: Representing scenes as neural radiance fields for view synthesis," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2020, pp. 405–421, doi: 10.1007/978-3-030-58452-8_24.

[58] S. J. Garbin, M. Kowalski, M. Johnson, J. Shotton, and J. Valentin, "FastNeRF: High-fidelity neural rendering at 200FPS," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 14326–14335, doi: 10.1109/ICCV48922.2021.01408.

[59] T. Neff et al., "DONeRF: Towards real-time rendering of compact neural radiance fields using depth Oracle networks," *Comput. Graph. Forum*, vol. 40, no. 4, pp. 45–59, Jul. 2021, doi: 10.1111/cgf.14340.

[60] C. Reiser, S. Peng, Y. Liao, and A. Geiger, "KiloNeRF: Speeding up neural radiance fields with thousands of tiny MLPs," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 14315–14325, doi: 10.1109/ICCV48922.2021.01407.

[61] K. Zhang, G. Riegler, N. Snavely, and V. Koltun, "NeRF++: Analyzing and improving neural radiance fields," 2020, *arXiv:2010.07492*.

[62] A. Yu, V. Ye, M. Tancik, and A. Kanazawa, "PixelNeRF: Neural radiance fields from one or few images," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 4576–4585, doi: 10.1109/CVPR46437.2021.00455.

[63] A. Chen et al., "MVSNeRF: Fast generalizable radiance field reconstruction from multi-view stereo," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 14104–14113, doi: 10.1109/ICCV48922.2021.01386.

[64] P. P. Srinivasan, B. Deng, X. Zhang, M. Tancik, B. Mildenhall, and J. T. Barron, "NeRV: Neural reflectance and visibility fields for relighting and view synthesis," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 7491–7500, doi: 10.1109/CVPR46437.2021.00741.

[65] M. Boss, R. Braun, V. Jampani, J. T. Barron, C. Liu, and H. P. A. Lensch, "NeRD: Neural reflectance decomposition from image collections," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 12664–12674, doi: 10.1109/ICCV48922.2021.01245.

[66] G. Gafni, J. Thies, M. Zollhöfer, and M. Nießner, "Dynamic neural radiance fields for monocular 4D facial avatar reconstruction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 8645–8654, doi: 10.1109/CVPR46437.2021.00854.

[67] K. Park et al., "Nerfies: Deformable neural radiance fields," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 5845–5854, doi: 10.1109/ICCV48922.2021.00581.

[68] E. Tretschk, A. Tewari, V. Golyanik, M. Zollhöfer, C. Lassner, and C. Theobalt, "Non-rigid neural radiance fields: Reconstruction and novel view synthesis of a dynamic scene from monocular video," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 12939–12950, doi: 10.1109/ICCV48922.2021.01272.

[69] H. Kato, Y. Ushiku, and T. Harada, "Neural 3D mesh renderer," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3907–3916, doi: 10.1109/CVPR.2018.00411.

[70] J. Pan, X. Han, W. Chen, J. Tang, and K. Jia, "Deep mesh reconstruction from single RGB images via topology modification networks," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 9963–9972, doi: 10.1109/ICCV.2019.01006.

[71] J. Tang, X. Han, J. Pan, K. Jia, and X. Tong, "A skeleton-bridged deep learning approach for generating meshes of complex topologies from single RGB images," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4536–4545, doi: 10.1109/CVPR.2019.00467.

[72] Q. Wang, Y. Tan, and Z. Mei, "Computational methods of acquisition and processing of 3D point cloud data for construction applications," *Arch. Comput. Methods Eng.*, vol. 27, no. 2, pp. 479–499, Apr. 2020, doi: 10.1007/s11831-019-09320-4.

[73] J. Shan and C. K. Toth, *Topographic Laser Ranging and Scanning: Principles and Processing*. Boca Raton, FL, USA: CRC Press, 2017, doi: 10.1201/9781315154381.

[74] Y. Gu, Q. Wang, X. Jia, and J. A. Benediktsson, "A novel MKL model of integrating LiDAR data and MSI for urban area classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 53, no. 10, pp. 5312–5326, Oct. 2015, doi: 10.1109/TGRS.2015.2421051.

[75] J. Fernandez-Diaz et al., "Capability assessment and performance metrics for the Titan multispectral mapping LiDAR," *Remote Sens.*, vol. 8, no. 11, p. 936, Nov. 2016, doi: 10.3390/rs8110936. [Online]. Available: https://www.mdpi.com/2072-4292/8/11/936

[76] M. Pedergnana, P. R. Marpu, M. D. Mura, J. A. Benediktsson, and L. Bruzzone, "Classification of remote sensing optical and LiDAR data using extended attribute profiles," *IEEE J. Sel. Topics Signal Process.*, vol. 6, no. 7, pp. 856–865, Nov. 2012, doi: 10.1109/JSTSP2012.2208177.

[77] M. Kukkonen, M. Maltamo, L. Korhonen, and P. Packalen, "Comparison of multispectral airborne laser scanning and stereo matching of aerial images as a single sensor solution to forest inventories by tree species," *Remote Sens. Environ.*, vol. 231, Sep. 2019, Art. no. 111208, doi: 10.1016/j.rse.2019.05.027. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0034425719302214

[78] M. A. Isa and I. Lazoglu, "Design and analysis of a 3D laser scanner," *Measurement*, vol. 111, pp. 122–133, Dec. 2017, doi: 10.1016/j.measurement.2017.07.028. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0263224117304633

[79] F. Rottensteiner et al., "The ISPRS benchmark on urban object classification and 3D building reconstruction," *ISPRS Ann. Photogramm., Remote Sens. Spatial Inf. Sci.*, vol. I-3, pp. 293–298, Jul. 2012, doi: 10.5194/isprsannals-i-3-293-2012.

[80] Wang, Chen, Zhu, Liu, Li, and Zheng, "A survey of mobile laser scanning applications and key techniques over urban areas," *Remote Sens.*, vol. 11, no. 13, p. 1540, Jun. 2019, doi: 10.3390/rs11131540. [Online]. Available: https://www.mdpi.com/2072-4292/11/13/1540

[81] M. Elhousni and X. Huang, "A survey on 3D LiDAR localization for autonomous vehicles," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Oct. 2020, pp. 1879–1884, doi: 10.1109/IV47402.2020.9304812.

[82] J. Li, B. Yang, Y. Cong, L. Cao, X. Fu, and Z. Dong, "3D forest mapping using a low-cost UAV laser scanning system: Investigation and comparison," *Remote Sens.*, vol. 11, no. 6, p. 717, Mar. 2019, doi: 10.3390/rs11060717.

[83] T. Zwęgliński, "The use of drones in disaster aerial needs reconnaissance and damage assessment—Three-dimensional modeling and orthophoto map study," *Sustainability*, vol. 12, no. 15, p. 6080, Jul. 2020, doi: 10.3390/su12156080. https://www.mdpi.com/2071-1050/12/15/6080

[84] F. Leberl et al., "Point clouds: LiDAR versus 3D vision," *Photogramm. Eng. Remote Sens.*, vol. 76, no. 10, pp. 1123–1134, Oct. 2010, doi: 10.14358/PERS.76.10.1123.

[85] Q. Hu, J. Luo, G. Hu, W. Duan, and H. Zhou, "3D point cloud generation using incremental structure-from-motion," *J. Phys., Conf. Ser.*, vol. 1087, Sep. 2018, Art. no. 062031, doi: 10.1088/1742-6596/1087/6/062031.

[86] S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski, "A comparison and evaluation of multi-view stereo reconstruction algorithms," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 1, Jun. 2006, pp. 519–528, doi: 10.1109/CVPR.2006.19.

[87] K. Kamal et al., "Performance assessment of Kinect as a sensor for pothole imaging and metrology," *Int. J. Pavement Eng.*, vol. 19, no. 7, pp. 565–576, Jul. 2018, doi: 10.1080/10298436.2016.1187730.

[88] C. Jia, T. Yang, C. Wang, B. Fan, and F. He, "A new fast filtering algorithm for a 3D point cloud based on RGB-D information," *PLoS ONE*, vol. 14, no. 8, Aug. 2019, Art. no. e0220253, doi: 10.1371/journal.pone.0220253.

[89] C. Chen, B. S. Yang, and S. Song, "Low cost and efficient 3D indoor mapping using multiple consumer RGB-D cameras," *Int. Arch. Photogramm., Remote Sens. Spatial Inf. Sci.*, vol. XLI-B1, pp. 169–174, Jun. 2016, doi: 10.5194/isprs-archives-XLI-B1-169-2016.

[90] H. Sarbolandi, D. Lefloch, and A. Kolb, "Kinect range sensing: Structured-light versus time-of-flight Kinect," *Comput. Vis. Image Understand.*, vol. 139, pp. 1–20, Oct. 2015, doi: 10.1016/j.cviu.2015.05.006.

[91] R. Bürgmann, P. A. Rosen, and E. J. Fielding, "Synthetic aperture radar interferometry to measure Earth's surface topography and its deformation," *Annu. Rev. Earth Planet. Sci.*, vol. 28, no. 1, pp. 169–209, May 2000, doi: 10.1146/annurev.earth.28.1.169.

[92] A. Ferretti, C. Prati, and F. Rocca, "Permanent scatterers in SAR interferometry," in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Jun. 1999, pp. 1528–1530, doi: 10.1109/IGARSS.1999.772008.

[93] A. Budillon, M. Crosetto, and O. Monserrat, "Editorial for the special issue 'urban deformation monitoring using persistent scatterer interferometry and SAR tomography,'" *Remote Sens.*, vol. 11, no. 11, p. 1306, May 2019, doi: 10.3390/rs11111306. https://www.mdpi.com/2072-4292/11/11/1306

[94] A. Gruen, "Fundamentals of videogrammetry—A review," *Hum. Movement Sci.*, vol. 16, no. 2, pp. 155–187, 1997, doi: 10.1016/S0167-9457(96)00048-6. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167945796000486

[95] A. Torresani and F. Remondino, "Videogrammetry VS photogrammetry for heritage 3D reconstruction," *Int. Arch. Photogramm., Remote Sens. Spatial Inf. Sci.*, vol. XLII-2/W15, pp. 1157–1162, Aug. 2019, doi: 10.5194/isprs-archives-XLII-2-W15-1157-2019.

[96] A. X. Chang et al., "ShapeNet: An information-rich 3D model repository," 2015, *arXiv:1512.03012*.

[97] K. Mo et al., "PartNet: A large-scale benchmark for fine-grained and hierarchical part-level 3D object

understanding," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 909–918, doi: 10.1109/CVPR.2019.00100.

[98] Z. Wu et al., "3D ShapeNets: A deep representation for volumetric shapes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1912–1920, doi: 10.1109/CVPR.2015.7298801.

[99] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pp. 3354–3361, Jun. 2012, doi: 10.1109/CVPR.2012.6248074. [Online]. Available: http://www.cvlibs.net/datasets/kitti/index.php

[100] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1231–1237, Sep. 2013, doi: 10.1177/0278364913491297. [Online]. Available: http://www.cvlibs.net/datasets/kitti/index.php

[101] J. Behley et al., "SemanticKITTI: A dataset for semantic scene understanding of LiDAR sequences," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 9297–9307, doi: 10.1109/ICCV.2019.00939. [Online]. Available: http://www.semantic-kitti.org/

[102] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "ScanNet: Richly-annotated 3D reconstructions of indoor scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2432–2443, doi: 10.1109/CVPR.2017.261.

[103] A. Chang et al., "Matterport3D: Learning from RGB-D data in indoor environments," in *Proc. Int. Conf. 3D Vis. (3DV)*, pp. 667–676, Oct. 2017, doi: 10.1109/3DV.2017.00081. [Online]. Available: https://niessner.github.io/Matterport/

[104] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from RGBD images," in *Eur. Conf. Comput. Vis.* Springer, 2012, pp. 746–760, doi: 10.1007/978-.3-642-33715-4_54. [Online]. Available: https://cs.nyu.edu/~silberman/datasets/nyu_depth_v2.html

[105] J. Xiao, A. Owens, and A. Torralba, "SUN3D: A database of big spaces reconstructed using SfM and object labels," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 1625–1632, doi: 10.1109/ICCV.2013.458. [Online]. Available: http://sun3d.cs.princeton.edu/

[106] S. Song, S. P. Lichtenberg, and J. Xiao, "SUN RGB-D: A RGB-D scene understanding benchmark suite," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 567–576, doi: 10.1109/CVPR.2015.7298655.

[107] A. Janoch et al., "A category-level 3-D object dataset: Putting the Kinect to work," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops (ICCV Workshops)*, Nov. 2011, pp. 141–165, doi: 10.1109/ICCVW.2011.6130382.

[108] M. De Deuge, A. Quadros, C. Hung, and B. Douillard, "Unsupervised feature learning for classification of outdoor 3D scans," in *Proc. Australas. Conf. Robot. Autom.*, vol. 2, 2013, p. 1. [Online]. Available: http://www.acfr.usyd.edu.au/papers/SydneyUrbanObjectsDataset.shtml

[109] S. Koch et al., "ABC: A big CAD model dataset for geometric deep learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pp. 9593–9603, Jun. 2019, doi: 10.1109/CVPR.2019.00983. [Online]. Available: https://cs.nyu.edu/~zhongshi/publication/abc-dataset/

[110] T. Hackel, N. Savinov, L. Ladicky, J. D. Wegner, K. Schindler, and M. Pollefeys, "Semantic3D.Net: A new large-scale point cloud classification benchmark," *ISPRS Ann. Photogramm., Remote Sens. Spatial Inf. Sci.*, vol. IV-1-W1, pp. 91–98, May 2017, doi: 10.5194/ISPRS-ANNALS-IV-1-W1-91-2017. [Online]. Available: https://www.semantic3d.net/view_dbase.php?chl=1&orderBy=name&orderStyle=ASC#download

[111] M. Kölle et al., "The Hessigheim 3D (H3D) benchmark on semantic segmentation of high-resolution 3D point clouds and textured meshes from UAV LiDAR and Multi-View-Stereo," 2021, *arXiv:2102.05346*.

[112] H. Fu et al., "3D-FRONT: 3D furnished rooms with layOuts and semaNTics," 2020, *arXiv:2011.09127*.

[113] H. Fu et al., "3D-FUTURE: 3D furniture shape with TextURE," 2020, *arXiv:2009.09633*.

[114] Q. Hu, B. Yang, S. Khalid, W. Xiao, N. Trigoni, and A. Markham, "Towards semantic segmentation of urban-scale 3D point clouds: A dataset, benchmarks and challenges," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 4975–4985. [Online]. Available: http://point-cloud-analysis.cs.ox.ac.uk/

[115] (2014). *The Stanford Computer Graphics Laboratory*. [Online]. Available: https://graphics.stanford.edu/data/3Dscanrep/

[116] G. Turk and B. Mullins. (2021). *Large Geometric Models Archive*. [Online]. Available: https://www.cc.gatech.edu/projects/large_models/

[117] Y. Rubner, C. Tomasi, and L. J. Guibas, "A metric for distributions with applications to image databases," in *Proc. 6th Int. Conf. Comput. Vis.*, Jan. 1998, pp. 59–66, doi: 10.1109/ICCV.1998.710701.

[118] Y. Rubner, "The Earth mover's distance as a metric for image retrieval," *Int. J. Comput. Vis.*, vol. 40, no. 2, pp. 99–121, 2000, doi: 10.1023/A:1026543900054.

[119] M. Tatarchenko, S. R. Richter, R. Ranftl, Z. Li, V. Koltun, and T. Brox, "What do single-view 3D reconstruction networks learn?" in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 3400–3409, doi: 10.1109/CVPR.2019.00352.

[120] O. Taubert, M. Götz, A. Schug, and A. Streit, "Loss scheduling for class-imbalanced image segmentation problems," in *Proc. 19th IEEE Int. Conf. Mach. Learn. Appl. (ICMLA)*, Dec. 2020, pp. 426–431, doi: 10.1109/ICMLA51294.2020.00073.

[121] S. Kullback and R. A. Leibler, "On information and sufficiency," *Ann. Math. Statist.*, vol. 22, no. 1, pp. 79–86, Mar. 1951, doi: 10.1214/AOMS/1177729694.

[122] D. Chen, X. Tian, Y. Shen, and M. Ouhyoung, "On visual similarity based 3D model retrieval," *Comput. Graph. Forum*, vol. 22, no. 3, pp. 223–232, Sep. 2003, doi: 10.1111/1467-8659.00669.

[123] H. Kato et al., "Differentiable rendering: A survey," 2020, *arXiv:2006.12057*.

[124] B. Curless and M. Levoy, "A volumetric method for building complex models from range images," in *Proc. 23rd Annu. Conf. Comput. Graph. Interact. Techn.*, Aug. 1996, pp. 303–312, doi: 10.1145/237170.237269.

[125] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3D surface construction algorithm," in *Proc. 14th Annu. Conf. Comput. Graph. Interact. Techn.*, Aug. 1987, pp. 163–169, doi: 10.1145/37401.37422.

[126] D. Maturana and S. Scherer, "VoxNet: A 3D convolutional neural network for real-time object recognition," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2015, pp. 922–928, doi: 10.1109/IROS.2015.7353481.

[127] N. Sedaghat, M. Zolfaghari, E. Amiri, and T. Brox, "Orientation-boosted voxel nets for 3D object recognition," 2016, *arXiv:1604.03351*.

[128] C. R. Qi, H. Su, M. Nießner, A. Dai, M. Yan, and L. J. Guibas, "Volumetric and multi-view CNNs for object classification on 3D data," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 5648–5656, doi: 10.1109/CVPR.2016.609.

[129] V. Hegde and R. Zadeh, "FusionNet: 3D object classification using multiple data representations," 2016, *arXiv:1607.05695*.

[130] M. Kazhdan, T. Funkhouser, and S. Rusinkiewicz, "Rotation invariant spherical harmonic representation of 3D shape descriptors," in *Proc. Symp. Geometry Process.*, vol. 6, pp. 156–164, doi: 10.2312/SGP/SGP03/156-165.

[131] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller, "Multi-view convolutional neural networks for 3D shape recognition," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 945–953, doi: 10.1109/ICCV.2015.114.

[132] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: 10.1162/neco.1997.9.8.1735.

[133] A. Kar, S. Tulsiani, J. Carreira, and J. Malik, "Category-specific object reconstruction from a single image," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1966–1974, doi: 10.1109/CVPR.2015.7298807.

[134] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," 2013, *arXiv:1312.6114*.

[135] D. P. Kingma and M. Welling, "An introduction to variational autoencoders," *Found. Trends Mach. Learn.*, vol. 12, no. 4, pp. 307–392, 2019, doi: 10.1561/2200000056.

[136] I. J. Goodfellow et al., "Generative adversarial networks," 2014, *arXiv:1406.2661*.

[137] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," 2015, *arXiv:1511.06434*.

[138] J. Wu, C. Zhang, X. Zhang, Z. Zhang, W. T. Freeman, and J. B. Tenenbaum, "Learning shape priors for single-view 3D completion and reconstruction," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 646–662, doi: 10.1007/978-3-030-01252-6_40.

[139] A. Dai, C. Diller, and M. Niessner, "SG-NN: Sparse generative neural networks for self-supervised scene completion of RGB-D scans," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 846–855, doi: 10.1109/cvpr42600.2020.00093.

[140] C. Choy, J. Gwak, and S. Savarese, "4D spatio-temporal ConvNets: Minkowski convolutional neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 3070–3079, doi: 10.1109/CVPR.2019.00319.

[141] S. Roth and S. R. Richter, "Matryoshka networks: Predicting 3D geometry via nested shape layers," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 1936–1944, doi: 10.1109/CVPR.2018.00207.

[142] D. Meagher, "Geometric modeling using Octree encoding," *Comput. Graph. Image Process.*, vol. 19, no. 2, pp. 129–147, Jun. 1982, doi: 10.1016/0146-664X(82)90104-6.

[143] C. L. Jackins and S. L. Tanimoto, "Oct-trees and their use in representing three-dimensional objects," *Comput. Graph. Image Process.*, vol. 14, no. 3, pp. 249–270, Nov. 1980, doi: 10.1016/0146-664X(80)90055-6.

[144] G. Riegler, A. O. Ulusoy, and A. Geiger, "OctNet: Learning deep 3D representations at high resolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 3577–3586, doi: 10.1109/CVPR.2017.701.

[145] C. Zach, T. Pock, and H. Bischof, "A globally optimal algorithm for robust TV-L^1 range image integration," in *Proc. IEEE 11th Int. Conf. Comput. Vis.*, Oct. 2007, pp. 1–8, doi: 10.1109/ICCV.2007.4408983.

[146] M. Firman, O. M. Aodha, S. Julier, and G. J. Brostow, "Structured prediction of unobserved voxels from a single depth image," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 5431–5440, doi: 10.1109/CVPR.2016.586.

[147] P.-S. Wang, Y. Liu, Y.-X. Guo, C.-Y. Sun, and X. Tong, "O-CNN: Octree-based convolutional neural networks for 3D shape analysis," *ACM Trans. Graph.*, vol. 36, no. 4, pp. 1–11, Aug. 2017, doi: 10.1145/3072959.3073608.

[148] R. Q. Charles, H. Su, M. Kaichun, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*,

Jul. 2017, pp. 652–660, doi: 10.1109/CVPR.2017.16.
[149] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 31. Red Hook, NY, USA: Curran Associates, 2017, pp. 5099–5108. [Online]. Available: https://papers.nips.cc/paper/2017/file/d8bf84be3800d12f74d8b05e9b89836f-Paper.pdf
[150] R. Klokov and V. Lempitsky, "Escape from cells: Deep Kd-networks for the recognition of 3D point cloud models," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 863–872, doi: 10.1109/ICCV.2017.99.
[151] Y. Shen, C. Feng, Y. Yang, and D. Tian, "Mining point cloud local structures by kernel correlation and graph pooling," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4548–4557, doi: 10.1109/CVPR.2018.00478.
[152] D. Rezende and S. Mohamed, "Variational inference with normalizing flows," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1530–1538. [Online]. Available: http://proceedings.mlr.press/v37/rezende15.html
[153] R. T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. Duvenaud, "Neural ordinary differential equations," 2018, *arXiv:1806.07366*.
[154] W. Grathwohl, R. T. Q. Chen, J. Bettencourt, I. Sutskever, and D. Duvenaud, "FFJORD: Free-form continuous dynamics for scalable reversible generative models," 2018, *arXiv:1810.01367*.
[155] L. Yu, X. Li, C.-W. Fu, D. Cohen-Or, and P.-A. Heng, "PU-Net: Point cloud upsampling network," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2790–2799, doi: 10.1109/CVPR.2018.00295.
[156] W. Yifan, S. Wu, H. Huang, D. Cohen-Or, and O. Sorkine-Hornung, "Patch-based progressive 3D point set upsampling," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 5951–5960, doi: 10.1109/CVPR.2019.00611.
[157] R. Cai, G. Yang, H. Averbuch-Elor, Z. Hao, S. Belongie, N. Snavely, and B. Hariharan, "Learning gradient fields for shape generation," in *Proc. Eur. Conf. Comput. Vis.*, vol. 12348, 2020, pp. 364–381, doi: 10.1007/978-3-030-58580-8_22.
[158] P. Guerrero, Y. Kleiman, M. Ovsjanikov, and N. J. Mitra, "PCPNet learning local shape properties from raw point clouds," *Comput. Graph. Forum*, vol. 37, pp. 75–85, May 2018, doi: 10.1111/cgf.13343.
[159] Y. Ben-Shabat and S. Gould, "DeepFit: 3D surface fitting via neural network weighted least squares," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2020, pp. 20–34, doi: 10.1007/978-3-030-58452-8_2.
[160] M. Atzmon, H. Maron, and Y. Lipman, "Point convolutional neural networks by extension operators," *ACM Trans. Graph.*, vol. 37, no. 4, pp. 1–12, Aug. 2018, doi: 10.1145/3197517.3201301.
[161] J. Mao, X. Wang, and H. Li, "Interpolated convolutional networks for 3D point cloud understanding," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 1578–1587, doi: 10.1109/ICCV.2019.00166.
[162] H. Zhao, L. Jiang, C.-W. Fu, and J. Jia, "PointWeb: Enhancing local neighborhood features for point cloud processing," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 5560–5568, doi: 10.1109/CVPR.2019.00571.
[163] S. Lan, R. Yu, G. Yu, and L. S. Davis, "Modeling local geometric structure of 3D point clouds using geo-CNN," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 998–1008, doi: 10.1109/CVPR.2019.00109.
[164] W. Wu, Z. Qi, and L. Fuxin, "PointConv: Deep convolutional networks on 3D point clouds," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 9613–9622, doi: 10.1109/CVPR.2019.00985.
[165] Y. Zhao, T. Birdal, J. E. Lenssen, E. Menegatti, L. Guibas, and F. Tombari, "Quaternion equivariant capsule networks for 3D point clouds," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2020, pp. 1–19, doi: 10.1007/978-3-030-58452-8_1.
[166] F. Engelmann, T. Kontogianni, and B. Leibe, "Dilated point convolutions: On the receptive field size of point convolutions on 3D point clouds," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 9463–9469, doi: 10.1109/ICRA40945.2020.9197503.
[167] M. Tatarchenko, J. Park, V. Koltun, and Q.-Y. Zhou, "Tangent convolutions for dense prediction in 3D," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3887–3896, doi: 10.1109/CVPR.2018.00409.
[168] H. Su et al., "SPLATNet: Sparse lattice networks for point cloud processing," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2530–2539, doi: 10.1109/CVPR.2018.00268.
[169] S. Shi et al., "PV-RCNN: Point-voxel feature set abstraction for 3D object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 10526–10535, doi: 10.1109/cvpr42600.2020.01054.
[170] M. Rakotosaona, V. La Barbera, P. Guerrero, N. J. Mitra, and M. Ovsjanikov, "PointCleanNet: Learning to denoise and remove outliers from dense point clouds," *Comput. Graph. Forum*, vol. 39, no. 1, pp. 185–203, Feb. 2020, doi: 10.1111/cgf.13753.
[171] E. J. Smith, S. Fujimoto, A. Romero, and D. Meger, "GEOMetrics: Exploiting geometric structure for graph-encoded objects," 2019, *arXiv:1901.11461*.
[172] J. K. Pontes, C. Kong, S. Sridharan, S. Lucey, A. Eriksson, and C. Fookes, "Image2Mesh: A learning framework for single image 3D reconstruction," in *Proc. Asian Conf. Comput. Vis.* Cham, Switzerland: Springer, 2018, pp. 365–381, doi: 10.1007/978-3-030-20887-5_23.
[173] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778, doi: 10.1109/cvpr.2016.90.
[174] F. Williams, T. Schneider, C. Silva, D. Zorin, J. Bruna, and D. Panozzo, "Deep geometric prior for surface reconstruction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 10122–10131, doi: 10.1109/CVPR.2019.01037.
[175] A. Kar, C. Häne, and J. Malik, "Learning a multi-view stereo machine," 2017, *arXiv:1708.05375*.
[176] R. Hanocka, A. Hertz, N. Fish, R. Giryes, S. Fleishman, and D. Cohen-Or, "MeshCNN: A network with an edge," *ACM Trans. Graph.*, vol. 38, no. 4, pp. 1–12, Aug. 2019, doi: 10.1145/3306346.3322959.
[177] D. Coquelin et al., "Accelerating neural network training with distributed asynchronous and selective optimization (DASO)," *J. Big Data*, vol. 9, no. 1, pp. 1–18, Dec. 2022, doi: 10.1186/s40537-021-00556-1.
[178] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2980–2988, doi: 10.1109/ICCV.2017.322.
[179] J. You, R. Ying, X. Ren, W. Hamilton, and J. Leskovec, "GraphRNN: Generating realistic graphs with deep auto-regressive models," in *Proc. Int. Conf. Mach. Learn.*, vol. 80, 2018, pp. 5708–5717. [Online]. Available: http://proceedings.mlr.press/v80/you18a/you18a.pdf
[180] A. Van Oord, N. Kalchbrenner, and K. Kavukcuoglu, "Pixel recurrent neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1747–1756. [Online]. Available: http://proceedings.mlr.press/v48/oord16.pdf
[181] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Netw.*, vol. 2, no. 5, pp. 359–366, Jan. 1989, doi: 10.1016/0893-6080(89)90020-8.
[182] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, "Surface reconstruction from unorganized points," in *Proc. 19th Annu. Conf. Comput. Graph. Interact. Techn.*, Jul. 1992, pp. 71–78, doi: 10.1145/133994.134011.
[183] E. Tretschk, A. Tewari, V. Golyanik, M. Zollhöfer, C. Stoll, and C. Theobalt, "PatchNets: Patch-based generalizable deep implicit 3D shape representations," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2020, pp. 293–309, doi: 10.1007/978-3-030-58517-4_18.
[184] K. Genova, F. Cole, A. Sud, A. Sarna, and T. Funkhouser, "Local deep implicit functions for 3D shape," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 4856–4865, doi: 10.1109/CVPR42600.2020.00491.
[185] J. Chibane and G. Pons-Moll, "Implicit feature networks for texture completion from partial 3D data," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2020, pp. 717–725, doi: 10.1007/978-3-030-66096-3_48.
[186] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Int. Conf. Med. Image Comput. Comput.-Assist. Intervent*. Cham, Switzerland: Springer, 2015, pp. 234–241, doi: 10.1007/978-3-319-24574-4_28.
[187] Ö. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger, "3D U-Net: Learning dense volumetric segmentation from sparse annotation," in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Intervent*. Cham, Switzerland: Springer, 2016, pp. 424–432, doi: 10.1007/978-3-319-46723-8_49.
[188] S. Saito, T. Simon, J. Saragih, and H. Joo, "PIFuHD: Multi-level pixel-aligned implicit function for high-resolution 3D human digitization," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 81–90, doi: 10.1109/cvpr42600.2020.00016.
[189] B. Lal Bhatnagar, C. Sminchisescu, C. Theobalt, and G. Pons-Moll, "Combining implicit function learning and parametric models for 3D human reconstruction," 2020, *arXiv:2007.11432*.
[190] M. Niemeyer, L. Mescheder, M. Oechsle, and A. Geiger, "Occupancy flow: 4D reconstruction by learning particle dynamics," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 5378–5388, doi: 10.1109/ICCV.2019.00548.
[191] S. Saito, Z. Huang, R. Natsume, S. Morishima, H. Li, and A. Kanazawa, "PIFu: Pixel-aligned implicit function for high-resolution clothed human digitization," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 2304–2314, doi: 10.1109/ICCV.2019.00239.
[192] M. Oechsle, L. Mescheder, M. Niemeyer, T. Strauss, and A. Geiger, "Texture fields: Learning texture representations in function space," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 4530–4539, doi: 10.1109/ICCV.2019.00463.
[193] V. Sitzmann, M. Zollhöfer, and G. Wetzstein, "Scene representation networks: Continuous 3D-Structure-Aware neural scene representations," 2019, *arXiv:1906.01618*.
[194] M. Niemeyer, L. Mescheder, M. Oechsle, and A. Geiger, "Differentiable volumetric rendering: Learning implicit 3D representations without 3D supervision," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 3501–3512, doi: 10.1109/CVPR42600.2020.00356.
[195] L. Yariv et al., "Multiview neural surface reconstruction by disentangling geometry and appearance," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 2492–2502. [Online]. Available: https://proceedings.neurips.cc/paper/2020/file/1a77befc3b608d6ed363567685f70e1e-Paper.pdf
[196] Q. Xu, W. Wang, D. Ceylan, R. Mech, and U. Neumann, "DISN: Deep implicit surface network for high-quality single-view 3D reconstruction," 2019, *arXiv:1905.10711*.
[197] S. Liu, Y. Zhang, S. Peng, B. Shi, M. Pollefeys, and

Z. Cui, "DIST: Rendering deep implicit signed distance function with differentiable sphere tracing," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 2016–2025, doi: 10.1109/CVPR42600.2020.00209.

[198] Y. Jiang, D. Ji, Z. Han, and M. Zwicker, "SDFDiff: Differentiable rendering of signed distance fields for 3D shape optimization," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 1248–1258, doi: 10.1109/cvpr42600.2020.00133.

[199] Y. Duan, H. Zhu, H. Wang, L. Yi, R. Nevatia, and L. J. Guibas, "Curriculum DeepSDF," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2020, pp. 51–67, doi: 10.1007/978-3-030-58598-3_4.

[200] T. Takikawa et al., "Neural geometric level of detail: Real-time rendering with implicit 3D shapes," 2021, *arXiv:2101.10994*.

[201] S. Duggal et al., "Mending neural implicit modeling for 3D vehicle reconstruction in the wild," 2021, *arXiv:2101.06860*.

[202] S. Liu, S. Saito, W. Chen, and H. Li, "Learning to infer implicit surfaces without 3D supervision," 2019, *arXiv:1911.00767*.

[203] E. Chatzipantazis, S. Pertigkiozoglou, E. Dobriban, and K. Daniilidis, "SE(3)-equivariant attention networks for shape reconstruction in function space," 2022, *arXiv:2204.02394*.

[204] F. Fuchs, D. Worrall, V. Fischer, and M. Welling, "SE(3)-transformers: 3D roto-translation equivariant attention networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 1970–1981.

[205] N. Thomas et al., "Tensor field networks: Rotation- and translation-equivariant neural networks for 3D point clouds," 2018, *arXiv:1802.08219*.

[206] C. Deng, O. Litany, Y. Duan, A. Poulenard, A. Tagliasacchi, and L. Guibas, "Vector neurons: A general framework for SO(3)-equivariant networks," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 12180–12189, doi: 10.1109/ICCV48922.2021.01198.

[207] Y. Chen, B. Fernando, H. Bilen, M. Nießner, and E. Gavves, "3D equivariant graph implicit functions," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2022, pp. 485–502.

[208] A. Vaswani et al., "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008. [Online]. Available: https://papers.nips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[209] Q. Wang et al., "IBRNet: Learning multi-view image-based rendering," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 4688–4697, doi: 10.1109/CVPR46437.2021.00466.

[210] J. T. Barron, B. Mildenhall, M. Tancik, P. Hedman, R. Martin-Brualla, and P. P. Srinivasan, "Mip-NeRF: A multiscale representation for anti-aliasing neural radiance fields," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 5835–5844, doi: 10.1109/ICCV48922.2021.00580.

[211] J. T. Barron, B. Mildenhall, D. Verbin, P. P. Srinivasan, and P. Hedman, "Mip-NeRF 360: Unbounded anti-aliased neural radiance fields," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2022, pp. 5470–5479, doi: 10.1109/CVPR52688.2022.00539.

[212] J. T. Barron, B. Mildenhall, D. Verbin, P. P. Srinivasan, and P. Hedman, "Zip-NeRF: Anti-aliased grid-based neural radiance fields," 2023, *arXiv:2304.06706*.

[213] T. Müller, A. Evans, C. Schied, and A. Keller, "Instant neural graphics primitives with a multiresolution hash encoding," *ACM Trans. Graph.*, vol. 41, no. 4, pp. 1–15, Jul. 2022, doi: 10.1145/3528223.3530127.

[214] S. Bi et al., "Neural reflectance fields for appearance acquisition," 2020, *arXiv:2008.03824*.

[215] X. Zhang, P. P. Srinivasan, B. Deng, P. Debevec, W. T. Freeman, and J. T. Barron, "NeRFactor: Neural factorization of shape and reflectance under an unknown illumination," *ACM Trans. Graph.*, vol. 40, no. 6, pp. 1–18, Dec. 2021, doi: 10.1145/3478513.3480496.

[216] Z. Wang, S. Wu, W. Xie, M. Chen, and V. Adrian Prisacariu, "NeRF–: Neural radiance fields without known camera parameters," 2021, *arXiv:2102.07064*.

[217] A. Pumarola, E. Corona, G. Pons-Moll, and F. Moreno-Noguer, "D-NeRF: Neural radiance fields for dynamic scenes," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 10313–10322, doi: 10.1109/CVPR46437.2021.01018.

[218] Z. Li, S. Niklaus, N. Snavely, and O. Wang, "Neural scene flow fields for space-time view synthesis of dynamic scenes," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 6494–6504, doi: 10.1109/CVPR46437.2021.00643.

[219] K. Park et al., "HyperNeRF: A higher-dimensional representation for topologically varying neural radiance fields," 2021, *arXiv:2106.13228*.

[220] P. Hedman, P. P. Srinivasan, B. Mildenhall, J. T. Barron, and P. Debevec, "Baking neural radiance fields for real-time view synthesis," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 5855–5864, doi: 10.1109/ICCV48922.2021.00582.

[221] K. Gao, Y. Gao, H. He, D. Lu, L. Xu, and J. Li, "NeRF: Neural radiance field in 3D vision, a comprehensive review," 2022, *arXiv:2210.00379*.

[222] P. Wang, L. Liu, Y. Liu, C. Theobalt, T. Komura, and W. Wang, "NeuS: Learning neural implicit surfaces by volume rendering for multi-view reconstruction," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 27171–27183. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2021/file/e41e164f7485ec4a28741a2d0ea41c74-Paper.pdf

[223] M. Oechsle, S. Peng, and A. Geiger, "UNISURF: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 5569–5579.

[224] Y. Wang, I. Skorokhodov, and P. Wonka, "HF-NeuS: Improved surface reconstruction using high-frequency details," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 35, 2022, pp. 1966–1978. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2022/file/0ce8e3434c7b486bbddff9745b2a1722-Paper-Conference.pdf

[225] Q. Fu, Q. Xu, Y. S. Ong, and W. Tao, "Geo-Neus: Geometry-consistent neural implicit surfaces learning for multi-view reconstruction," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 35, 2022, pp. 3403–3416. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2022/file/16415eed5a0a121bfce79924db05d3fe-Paper-Conference.pdf

[226] L. Yariv, J. Gu, Y. Kasten, and Y. Lipman, "Volume rendering of neural implicit surfaces," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 4805–4815. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2021/file/25e2a30f44898b9f3e978b1786dcd85c-Paper.pdf

[227] Z. Yu et al. (2022). *SDFStudio: A Unified Framework for Surface Reconstruction*. [Online]. Available: https://github.com/autonomousvision/sdfstudio

[228] M. Tancik et al., "Nerfstudio: A modular framework for neural radiance field development," in *Proc. Special Interest Group Comput. Graph. Interact. Techn. Conf. Conf.*, Jul. 2023, pp. 1–12, doi: 10.1145/3588432.3591516.

[229] Y. Xiang, R. Mottaghi, and S. Savarese, "Beyond Pascal: A benchmark for 3D object detection in the wild," in *Proc. IEEE Winter Conf. Appl. Comput. Vis.*, Mar. 2014, pp. 75–82, doi: 10.1109/WACV.2014.6836101.

[230] H. O. Song, Y. Xiang, S. Jegelka, and S. Savarese, "Deep metric learning via lifted structured feature embedding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 4004–4012, doi: 10.1109/CVPR.2016.434.

[231] J. J. Lim, H. Pirsiavash, and A. Torralba, "Parsing IKEA objects: Fine pose estimation," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 2992–2999, doi: 10.1109/ICCV.2013.372.

[232] S. Choi, Q.-Y. Zhou, S. Miller, and V. Koltun, "A large dataset of object scans," 2016, *arXiv:1602.02481*.

[233] Trimble Inc. (2023). *3D Warehouse*. [Online]. Available: https://3dwarehouse.sketchup.com/

[234] R. Jensen, A. Dahl, G. Vogiatzis, E. Tola, and H. Aanæs, "Large scale multi-view stereopsis evaluation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 406–413.

[235] Y. Yao et al., "BlendedMVS: A large-scale dataset for generalized multi-view stereo networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 1787–1796.

[236] J. McCormac, A. Handa, S. Leutenegger, and A. J. Davison, "SceneNet RGB-D: Can 5M synthetic images beat generic ImageNet pre-training on indoor segmentation?" in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2697–2706, doi: 10.1109/ICCV.2017.292.

[237] A. Bozic, P. Palafox, J. Thies, A. Dai, and M. Niessner, "TransformerFusion: Monocular RGB scene reconstruction using transformers," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2021, pp. 1403–1414. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2021/file/0a87257e5308197df43230edf4ad1dae-Paper.pdf

[238] H. Zhao, L. Jiang, J. Jia, P. Torr, and V. Koltun, "Point transformer," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, pp. 16239–16248, Oct. 2021. [Online]. Available: https://openaccess.thecvf.com/content/ICCV2021/papers/Zhao_Point_Transformer_ICCV_2021_paper.pdf

[239] M.-H. Guo, J.-X. Cai, Z.-N. Liu, T.-J. Mu, R. R. Martin, and S.-M. Hu, "PCT: Point cloud transformer," *Comput. Vis. Media*, vol. 7, no. 2, pp. 187–199, Jun. 2021, doi: 10.1007/s41095-021-0229-5.

[240] Y. LeCun and I. Misra. (2021). *Self-Supervised Learning: The Dark Matter of Intelligence*. [Online]. Available: https://ai.facebook.com/blog/self-supervised-learning-the-dark-matter-of-intelligence/

[241] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli, "Deep unsupervised learning using nonequilibrium thermodynamics," in *Proc. Int. Conf. Mach. Learn.*, vol. 37, 2015, pp. 2256–2265. [Online]. Available: http://proceedings.mlr.press/v37/sohl-dickstein15.pdf

[242] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, Eds. Red Hook, NY, USA: Curran Associates, 2020, pp. 6840–6851. [Online]. Available: https://proceedings.neurips.cc/paper/2020/file/4c5bcfec8584af0d967f1ab10179ca4b-Paper.pdf

[243] A. Q. Nichol and P. Dhariwal, "Improved denoising diffusion probabilistic models," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 8162–8171. [Online]. Available: http://proceedings.mlr.press/v139/nichol21a/nichol21a.pdf

[244] P. Dhariwal and A. Nichol, "Diffusion models beat GANs on image synthesis," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 8780–8794. [Online]. Available: https://proceedings.neurips.cc/paper/2021/file/49ad23d1ec9fa4bd8d77d02681df5cfa-Paper.pdf

[245] S. Luo and W. Hu, "Diffusion probabilistic models for 3D point cloud generation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 2836–2844, doi: 10.1109/CVPR46437.2021.00286.

[246] L. Zhou, Y. Du, and J. Wu, "3D shape generation and completion through point-voxel diffusion," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 5806–5815, doi: 10.1109/ICCV48922.2021.00577.

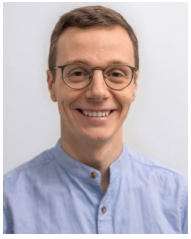
ABOUT THE AUTHORS

Anis Farshian received the B.Sc. degree in computer engineering from the Technical Faculty, Shariati Technical and Vocational College, Tehran, Iran, in 2012, and the M.Sc. degree in information technology engineering from the Iran University of Science and Technology, Tehran, in 2017.

She is currently with the Steinbuch Centre for Computing, Karlsruhe Institute of Technology, Karlsruhe, Germany. As part of her research and her membership in the Helmholtz Analytics Framework project and the Helmholtz AI project, she is working on methods for surface reconstruction for applied scientific problems. Her research interests include 3-D analysis, machine learning, and computer-assisted health.

Markus Götz (Member, IEEE) received the B.Sc. and M.Sc. degrees in IT-system engineering from the University of Potsdam, Potsdam, Germany, in 2010 and 2014, respectively, with intermediate stays at the Blekinge Tekniska Högskola, Karlskrona, Sweden, and CERN, Meyrin, Switzerland. He is currently working toward the Ph.D. degree in computational engineering at the University of Iceland, Reykjavik, Iceland, in conjunction with the Juelich Supercomputing Centre, Jülich, Germany.

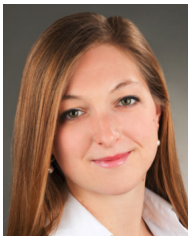
He is currently a Postdoctoral Researcher with the Steinbuch Centre for Computing, Karlsruhe Institute of Technology, Karlsruhe, Germany, where he is also the Project Manager for the Helmholtz Analytics Framework and the Head of the Helmholtz AI Consultants Team. In line with his work, he focuses on applied artificial intelligence and data analysis on high-performance cluster systems to work on the grand challenges in the natural sciences. His research interests include machine learning, global optimization, and parallel algorithm engineering.



Gabriele Cavallaro (Senior Member, IEEE) received the B.Sc. and M.Sc. degrees in telecommunications engineering from the University of Trento, Trento, Italy, in 2011 and 2013, respectively, and the Ph.D. degree in electrical and computer engineering from the University of Iceland, Reykjavik, Iceland, in 2016.

From 2016 to 2021, he was the Deputy Head of the High Productivity Data Processing (HPDP) Research Group, Jülich Supercomputing Centre (JSC), Forschungszentrum Jülich, Jülich, Germany. Since 2022, he has been the Head of the "AI and ML for Remote Sensing" Simulation and Data Laboratory, JSC, and an Adjunct Associate Professor with the School of Natural Sciences and Engineering, University of Iceland. Concurrently, he serves as a Visiting Professor at the Φ-Laboratory, European Space Agency (ESA), Italy, where he contributes to the Quantum Computing for Earth Observation (QC4EO) initiative. His research interests cover remote sensing data processing with parallel machine learning algorithms that scale on distributed computing systems and innovative computing technologies.

Dr. Cavallaro was a recipient of the IEEE GRSS Third Prize in the Student Paper Competition of the IEEE International Geoscience and Remote Sensing Symposium (IGARSS) 2015 (Milan, Italy). From 2020 to 2023, he was the Chair of the High-Performance and Disruptive Computing in Remote Sensing (HDCRS) Working Group under the IEEE GRSS Earth Science Informatics Technical Committee (ESI TC). In 2023, he took on the role of Co-Chair of the ESI TC. He has been serving as an Associate Editor for IEEE TRANSACTIONS ON IMAGE PROCESSING (TIP) since October 2022.



Charlotte Debus (Member, IEEE) studied physics at the University of Heidelberg, Heidelberg, Germany. She received the M.Sc. degree from the University of Heidelberg in 2012 and the Ph.D. degree in medical physics from the German Cancer Research Center (DKFZ), Heidelberg, in 2016.

After a two-year postdoctoral period at DKFZ, she was a Research Associate with the German Aerospace Institute, Cologne, Germany, from 2019 to 2020, where she worked on machine learning methods and high-performance computing. Since October 2020, she has been a Postdoctoral Researcher at the Helmholtz AI Local Energy Consultants Team, Steinbuch Centre for Computing, Karlsruhe Institute of Technology, Karlsruhe, Germany.

Matthias Nießner received the Ph.D. degree from the University of Erlangen-Nuremberg, Erlangen, Germany, in 2013.

He was a Visiting Assistant Professor with Stanford University, Stanford, CA, USA, from 2013 to 2017. Since 2017, he has been a Professor with the Technical University of Munich (TUM), Munich, Germany, focusing on cutting-edge research at the intersection of computer vision, graphics, and machine learning. He is currently the Head of the Visual Computing Laboratory, TUM. In addition to his academic career, he is a Co-Founder and the Director of Synthesia Inc., London, U.K., a startup empowering storytellers with artificial intelligence (AI). He is particularly interested in novel techniques for 3-D reconstruction, semantic 3-D scene understanding, and video editing.

Prof. Nießner is a TUM-IAS Rudolph Moessbauer Fellow. He received the Google Faculty Award for Machine Perception in 2017, the Nvidia Professor Partnership Award in 2018, and the ERC Starting Grant in 2018.

Jón Atli Benediktsson (Fellow, IEEE) received the Cand.Sci. degree in electrical engineering from the University of Iceland, Reykjavik, Iceland, in 1984, and the M.S.E.E. and Ph.D. degrees in electrical engineering from Purdue University, West Lafayette, IN, USA, in 1987 and 1990, respectively.

From 2009 to 2015, he was a Pro-Rector of science and academic affairs and a Professor of electrical and computer engineering with the University of Iceland. Since July 2015, he has been the President and the Rector of the University of Iceland. He is a Co-Founder of the biomedical startup company Oxymap, Reykjavik. His research interests are in remote sensing, biomedical analysis of signals, pattern recognition, image processing, and signal processing. He has published extensively in these fields.

Dr. Benediktsson is a fellow of the Optics and Photonics Society (SPIE). He was a member of the 2014 IEEE Fellow Committee. He is a member of the Academia Europea, the Association of Chartered Engineers in Iceland (VFI), Societas Scientiarum Islandica, and Tau Beta Pi. He received the Stevan J. Kristof Award from Purdue University in 1991 as an outstanding graduate student in remote sensing. He was a recipient of the Icelandic Research Council's Outstanding Young Researcher Award in 1997.

In 2000, he was granted the IEEE Third Millennium Medal. In 2004, he was a co-recipient of the University of Iceland's Technology Innovation Award. In 2006, he received the Yearly Research Award from the Engineering Research Institute of the University of Iceland. He received the Outstanding Service Award from the IEEE Geoscience and Remote Sensing Society (GRSS) in 2007, the IEEE GRSS Education Award in 2020, the IEEE GRSS David Landgrebe Award in 2018, and the OECE Award from the School of Electrical and Computer Engineering (ECE), Purdue University, in 2016. He was co-recipient of the 2012 IEEE Transactions on Geoscience and Remote Sensing Paper Award. In 2013, he was a co-recipient of the IEEE GRSS Highest Impact Paper Award. In 2013, he received the IEEE/VFI Electrical Engineer of the Year Award. In 2016 and 2018, he was a co-recipient of the International Journal of Image and Data Fusion Best Paper Awards. In 2021, he was honored as a recipient of the Order of the Falcon from the President of Iceland. He was a Highly Cited Researcher (Clarivate Analysis) from 2018 to 2020. He was the 2011–2012 President of the IEEE GRSS and has been on the GRSS AdCom since 2000. He was the Editor-in-Chief of IEEE Transactions on Geoscience and Remote Sensing (TGRS) from 2003 to 2008. He has been serving as an Associate Editor of TGRS since 1999, IEEE Geoscience and Remote Sensing Letters since 2003, and IEEE Access since 2013. He was on the Editorial Board of the Proceedings of the IEEE from 2015 to 2020. He is on the International Editorial Board of *International Journal of Image and Data Fusion* and the Editorial Board of *Remote Sensing*. He was the Chairperson of the Steering Committee of IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing (J-STARS) from 2007 to 2010.

Achim Streit received the Diploma degree in computer science from the University of Dortmund, Dortmund, Germany, in 1999, and the Ph.D. degree in computer science from the University of Paderborn, Paderborn, Germany, in 2003.

He is one of the directors of the Steinbuch Centre for Computing, Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany. He is also a Professor of distributed and parallel high-performance computing systems with the Department of Informatics, KIT. Afterward, he led the Federated Systems and Data Division, Jülich Supercomputing Centre, Jülich, Germany. He initiated and chaired several national and international research initiatives within the Helmholtz Association [e.g., Helmholtz Data Federation and Helmholtz Information & Data Science Academy (HIDA)] on the national level (e.g., NFDI4Ing and NFDI-MatWerk) and the European level (e.g., EUDAT and EOSC). His research interests include high-performance and data-intensive computing, big data and federated data management, data analytics, job scheduling, and resource management for parallel and distributed systems.